

AFRL-IF-WP-TR-2002-1529

**ELECTRONIC DESIGN
AUTOMATION (EDA) ROADMAP
TASKFORCE REPORT
Design of Microprocessors**

**Silicon Integration Initiative (Si2)
Electronic Design Automation Industry Council**

**Silicon Integration Initiative, Inc.
4030 Braker Lane
Suite 550
Austin, TX 78759**



APRIL 1999

FINAL REPORT FOR 14 SEPTEMBER 1993 – 30 APRIL 1999

Approved for public release; distribution is unlimited.

**INFORMATION DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7334**

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE US GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS REPORT IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONS.

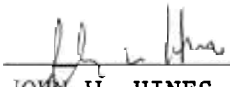
THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.



MAYA RUBEIZ, Project Engineer
Embedded Information Sys. Eng. Branch
Information Technology Division
Air Force Research Laboratory



JAMES S. WILLIAMSON, Chief
Embedded Information Sys. Eng. Branch
Information Technology Division
Air Force Research Laboratory



JOHN W. HINES, Actg Chief
Wright Site
Information Directorate

Do not return copies of this report unless contractual obligations or notice on a specific document requires its return.

REPORT DOCUMENTATION PAGE					<i>Form Approved</i> OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.						
1. REPORT DATE (DD-MM-YY) April 1999		2. REPORT TYPE Final		3. DATES COVERED (From - To) 09/14/1993 – 04/30/1999		
4. TITLE AND SUBTITLE ELECTRONIC DESIGN AUTOMATION (EDA) ROADMAP TASKFORCE REPORT Design of Microprocessors				5a. CONTRACT NUMBER F33615-93-C-1314		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER 63739E		
6. AUTHOR(S) Silicon Integration Initiative (Si2) Electronic Design Automation Industry Council				5d. PROJECT NUMBER A268		
				5e. TASK NUMBER 02		
				5f. WORK UNIT NUMBER 10		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Silicon Integration Initiative, Inc. 4030 Braker Lane Suite 550 Austin, TX 78759				8. PERFORMING ORGANIZATION REPORT NUMBER		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> Information Directorate Air Force Research Laboratory Air Force Materiel Command Wright-Patterson AFB, OH 45433-7334 </div> <div style="width: 45%;"> Defense Advanced Research Projects Agency Information Technology Office 3701 North Fairfax Drive Arlington, VA 22209-2308 </div> </div>				10. SPONSORING/MONITORING AGENCY ACRONYM(S) AFRL/IFTA		
				11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S) AFRL-IF-WP-TR-2002-1529		
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.						
13. SUPPLEMENTARY NOTES Report contains color.						
14. ABSTRACT The goal of this project was to support the establishment of tool interoperability standards for the semiconductor industry. Accomplishments include the publication of the “EDA Industry Standards Roadmap – 1996” and the “EDA Roadmap Taskforce Report – Design of Microprocessors.”						
15. SUBJECT TERMS Electronic Design Automation, EDA, EDA Industry Standards Roadmap, EDA Roadmap Taskforce Report, CAD Framework Initiative (CFI)						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT: SAR	18. NUMBER OF PAGES 190	19a. NAME OF RESPONSIBLE PERSON (Monitor) Maya Rubeiz 19b. TELEPHONE NUMBER (Include Area Code) (937) 255-6653 x3593	
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified				

TABLE OF CONTENTS

Section	Page
Introduction.....	1
EDA Roadmap Taskforce Report - Design of Microprocessors.....	2
Appendix - EDA Industry Standards Roadmap - 1996.....	A1

INTRODUCTION

Under the Rapid Prototyping of Application Specific Signal Processors (RASSP) Framework Integration Standards program, Silicon Integration Initiative (Si2), then known as CAD Framework Initiative (CFI), was directed to analyze industry feedback and support the Electronic Design Automation (EDA) Industry Council, both technically and administratively, in the development and publication of the EDA Standards Roadmap.

The final report delivered under this contract is the “EDA Roadmap Taskforce Report - Design of Microprocessors”. This document is an attempt at projecting the technical changes required to meet the needs of the semiconductor design community through the year 2003.

The “EDA Industry Standards Roadmap” was first published in 1996, and was again revised in 1998. This document analyzes many industry standards and recommends solutions to some of the major challenges of the next decade.

The “EDA Industry Standards Roadmap - 1996 has been included as an appendix to this document.



Electronic Design Automation Industry Council

EDA Roadmap Taskforce Report

Design of Microprocessors

Date of issue: 3/28/99

Published by: Si2, Inc.

©1999 by Silicon Integration Initiative, Inc. and Electronic Design Automation Industry Council

All rights reserved worldwide. Permission to use and reproduce this documentation for internal purposes is granted under the following conditions: no right is granted to modify, transmit, externally distribute or resell this documentation for commercial purposes. All reproductions shall include this copyright notice.

Table of Contents

TABLE OF CONTENTS.....	II
LIST OF FIGURES	IV
TASK FORCE RECOMMENDATIONS.....	V
FOREWORD.....	VI
ACKNOWLEDGEMENTS.....	VIII
SECTION 1	1
THE CHALLENGE TO EDA DESIGN AND TEST	1
TARGET DESIGN	3
<i>Year 2003 - Microprocessor Data Sheet</i>	<i>3</i>
SECTION 2	6
POWER.....	6
SECTION 3	10
SIGNAL INTEGRITY AND DELAY VARIATION.....	10
SECTION 4	14
DESIGN PRODUCTIVITY	14
<i>PRODUCTIVITY FACTORS</i>	<i>14</i>
<i>GUIDELINES</i>	<i>14</i>
<i>MEET-IN-THE-MIDDLE</i>	<i>15</i>
<i>AVOID PROBLEMS</i>	<i>16</i>
<i>ESTIMATION.....</i>	<i>16</i>
<i>SIMPLIFY THE DESIGN PROCESS.....</i>	<i>17</i>
<i>MODELING</i>	<i>18</i>
<i>FORECASTING</i>	<i>18</i>
SECTION 5	21
TYPES OF TESTING.....	21
ELECTRICAL TESTING CHALLENGES	21
PHYSICAL AND THERMAL CHALLENGES	21
ECONOMIC CHALLENGES.....	21
TEST YIELD CHALLENGES	22
BURN-IN AND RELIABILITY CHALLENGES	23
TIME-TO-VOLUME CHALLENGES.....	24
DEFECT IDENTIFICATION THROUGH ACCELERATED STRESS TESTING.....	24
<i>SHIFT TO BUILT-IN-SELF-TEST (BIST)</i>	<i>24</i>
<i>DESIGN TESTS DURING CHIP DESIGN.....</i>	<i>25</i>
<i>DEVELOP NEW FAULT MODELS & PLAN TESTS TO THEM</i>	<i>25</i>
SECTION 6	26
EDA SYSTEM STRUCTURE.....	26
SECTION 7	29

A VISION OF 100 NM MICROPROCESSOR DESIGN.....	29
PROCESS ENHANCEMENTS.....	29
METHODOLOGY ENHANCEMENTS.....	30
ELECTRONIC DESIGN AUTOMATION.....	32
SECTION 8.....	35
FUTURE RESEARCH	35
SOFT-ERRORS.....	35
ASYNCHRONOUS DESIGN.....	37
MEASURE DESIGN EFFECTIVENESS.....	37
SECTION 9.....	39
FUTURE TASKFORCE ACTIVITIES	39
APPENDIX A: ACRONYMS.....	A-1
APPENDIX B: REFERENCES.....	B-1

LIST OF FIGURES

FIGURE 1: TASKFORCE PROCESS - FOCUS ON CHANGE	1
FIGURE 2: TECHNOLOGY TRENDS.....	2
FIGURE 3: PUSHING FREQUENCY THROUGH PROCESS AND DESIGN CHANGES.....	2
FIGURE 4: POWER RISES SHARPLY.....	3
FIGURE 5: TARGET CHIP DATA SHEET.....	4
FIGURE 6: GLOBAL AND LOCAL INTERCONNECT DELAYS VS. GATE DELAYS.....	5
FIGURE 7: GROWTH IN ACTIVE CAPACITANCE LEADING TO POWER GROWTH.....	6
FIGURE 8: SUPPLY CURRENT.....	7
FIGURE 9: POWER / VOLTAGE = CURRENT.....	7
FIGURE 10: GATE / INTERCONNECT DELAY.....	10
FIGURE 11: SIGNAL INTEGRITY AND DELAY VARIATION.....	10
FIGURE 12: DELAY VARIATION.....	11
FIGURE 13: NEW SIGNAL INTERRELATIONSHIPS.....	12
FIGURE 14: INTERCONNECT-CENTRIC DESIGN SYSTEM.....	17
FIGURE 15: FORECASTING	19
FIGURE 16: AUTOMATED MODEL BUILDER	19
FIGURE 17: TEST COST IMPACT ON PRODUCT PRICING.....	22
FIGURE 18: YIELD LOSS DUE TO GUARD BANDING.....	22
FIGURE 19: COST PER BURN-IN SOCKET POSITION	23
FIGURE 20: TIME TO MARKET AND VOLUME INCREASING.....	23
FIGURE 21: TEMPERATURE AND VOLTAGE STRESS IDENTIFIES FAULTY BEHAVIOR.....	24
FIGURE 22: 6.1 FILE CENTRIC EDA.....	26
FIGURE 23: API CENTRIC EDA.....	27
FIGURE 24: INTERCONNECT-CENTRIC EDA.....	33
FIGURE 25: CMOS CHARGE IS DECREASING	36
FIGURE 26: SOFT ERRORS IN CMOS AND SOI.....	36

Task Force Recommendations

RECOMMENDATION I: POWER.....	9
RECOMMENDATION II: SIGNAL INTEGRITY AND DELAY UNCERTAINTY	13
RECOMMENDATION III: GUIDING PRINCIPLES	15
RECOMMENDATION IV: MEET IN THE MIDDLE DESIGN APPROACH.....	16
RECOMMENDATION V: ADDITIONAL GUIDELINES	18
RECOMMENDATION VI: PRODUCTIVITY.....	20
RECOMMENDATION VII: TEST.....	25
RECOMMENDATION VIII: EDA SYSTEM STRUCTURE	28
RECOMMENDATION IX: PROCESS MODIFICATIONS.....	30
RECOMMENDATION X: NEW DESIGN METHODOLOGIES	31
RECOMMENDATION XI: NEW EDA	34
RECOMMENDATION XII: FURTHER RESEARCH	38
RECOMMENDATION XIII: FUTURE TASKFORCE ACTIVITY	39

FOREWORD

This is the Report of the EDA Roadmap Taskforce on the Design of Microprocessors. This Roadmap is first in a series that attempts to project the future of technology for the design of electronic systems, particularly semiconductor integrated circuits. Starting with the design practices and tools in general use in 1998, it projects the changes required to meet the needs of the design community five years in the future, i.e. in 2003.

It is built on the National Technology Roadmap for Semiconductors 97 (NTRS97)[1]. We strongly recommend that readers of this report be familiar with NTRS97, it can be read or downloaded from the www.sematec.org[4] web site.

NTRS97 is the latest in a series that projects the evolution of semiconductor processing technology for five years into the future from date of issue. These Roadmaps have been remarkably accurate even while facing the formidable challenge of making projections in a technological field noted for its astounding rate of development. The Roadmaps are aggressive, making projections which at the time of issue seem futuristic — yet, in practice their forecasts are more often exceeded than not.

The NTRS97 authors approach their forecasting challenge in a unique manner, which futurists in other disciplines could well emulate. They set a stretch but attainable goal of an overall end result —an electronic system on a semiconductor chip which has defined capabilities well beyond the state of the art at the time. They then specify the evolution of each technology that must be attained in order to reach that goal. Some developments may take more effort than forecast, while others less. It is assumed that the worldwide semiconductor industrial, vendor, and university community will shift resources as needed to bring the lagging technologies through on time. Hence, the study is not a crystal-ball-gazing exercise, but a rigorous, schedulable plan of action to attain the goal.

NTRS97 identified *Design and Test* as a key technology whose evolution must be accelerated if the overall goals are to be reached. Readers are particularly directed to that chapter of the report.

This EDA Roadmap takes up the challenge of NTRS97 by digging deeper into Design and Test issues attempting to identify areas of critical R&D need or major design and test paradigm shifts.

This Roadmap is sponsored by the EDA (Electronic Design Automation) Industry Council[2]. The Industry Council is a committee that comprises EDA suppliers including EDAC[3], EDA users from both system companies, and semiconductor companies, as well as industry organizations including SEMATECH[4], SRC[5], and Si2[6]. This Taskforce was funded by DARPA[7] and Si2 administrated the work.

The EDA Industry Council chartered the EDA Roadmap Taskforce to select one electronic marketplace and examine changes that are required to design effectively five years hence. Implied by the Industry Council is the worry that the industry is unprepared for the future. Their hope is that an early prediction of needs and capabilities will allow enough lead-time for the development of the required advancements.

The EDA Roadmap Taskforce first met in February 1998. The Taskforce is comprised of 30 experts with semiconductor processing, design, or EDA backgrounds. They included representatives from both commercial vendors and customers of EDA. Many of the participants have a history in more than one of these fields. They examined several target markets to study (telecom, PC, consumer electronics, automotive, etc.) and chose High-end Microprocessor design as the most challenging. We expected that other taskforces would concentrate on such markets as telecom, PCs, consumer electronics, etc. The Taskforce then targeted designs that will begin in the year 2003, the time frame when semiconductor processes are forecasted to allow feature sizes smaller than one hundred nanometers.

The EDA Roadmap Taskforce met six times during 1998. It studied the NTRS97 report in detail. It obtained EDA information from many sources, including extrapolations from public plans that are published by commercial EDA companies. Much of the design information that the Taskforce obtained is anecdotal and it is not publishable. All anecdotal data was verified by determining if similar sources gave equivalent responses. In fact, the "stories" are surprisingly similar.

Investigation began at points where design paradigm shifts will or must occur. This included forecasting directions of semiconductor processing, design approaches, and electronic design automation.

Next, the Taskforce examined alternative solutions for identified paradigm shifts. The solutions included modification of semiconductor processing and design methodology, and R&D on new EDA tools. The report to follow contains the results of this work.

ACKNOWLEDGEMENTS

The Taskforce was comprised of highly experienced technologists. They strove to advance our science. Each member's affiliation indicates the member's association at the time that the member joined the committee.

Paul Weil, Co-Chair, Intel	Andrew B. Kahng, UCLA
Arny Goldfein, Co-Chair, Cadence	Sani Nassif, IBM
Shakhar Borkar, Intel	A. Richard Newton, UCB
Pat Capapano, Motorola	Bryan Preas, Xerox
Jason Cong, UCLA	Steve Sapiro, Consultant
Don Cottrell, Si2	Chuck Shaw, Cadence
Carlos Dangelo, LSILogic	Peter Suaris, Mentor
Steve Grout, SEMATECH	Rob Smith, Actel
Bill Joyner, SRC	

The Taskforce included the assistance from many other individuals. We would like to recognize the following for their special contributions to this effort. Each individual's affiliation is shown at the time of joining the study. We apologize for any names that we might have omitted in error.

Andy Charnas, Sun	Farid Najm, Univ of Ill
John Cohn, IBM	Phil Nigh, IBM
Mark Birnbaum, Fujitsu	Hillel Ofek, Dynalogic
Phil Fisher, SEMATECH	Saila Ponnappalli, IBM
Mark Flomenhoft, Intel	Larry Rosenberg, VSI
Randy Harr, Synopsys	Naresh Sehgal, Intel
Bernd Koenemann, LogicVision	Dick Smith, Cadence
Bob Lashley, Sun	Ron Walther, IBM

SECTION 1

The Challenge to EDA Design and Test

The ability to manufacture a semiconductor chip with certain capabilities is the driving force in electronic system technology. The evolution of that capability is the subject of the SIA National Technology Roadmap for Semiconductors '97 (NTRS97)[1] and it forecasts semiconductor fabrication capabilities in 2003 and beyond.

Integrated Circuit chips cannot be manufactured unless they can first be designed. NTRS97 identified Design and Test (D&T) as a critical technology whose evolution risked not being able to meet the challenges of designing the chips that could be fabricated in 2003. This EDA Roadmap Task took up the challenge of defining how D&T must evolve to prevent it from becoming the bottleneck, thus limiting the growth of the entire electronics industry.

The Taskforce is concerned that new EDA tools may not be ready for these new designs. Chip designs will start before the end of 2001. A gap may initially exist between EDA capabilities and the chip designers' needs. Among the goals of the Taskforce, is to focus EDA R&D efforts on the most critical problem areas.

It was concluded that without additional research in a number of areas, the required changes would probably never be made, compounding the problem. We hope that this will be thoroughly studied and debated throughout the industry, and that resources will be allocated to accelerate necessary activities to bring it to fruition.

To scope the problem, the Taskforce studied needs for high-end microprocessor design starts that will begin in 2003 — the time frame when semiconductor processes are forecasted to allow feature sizes smaller than one hundred nanometers.

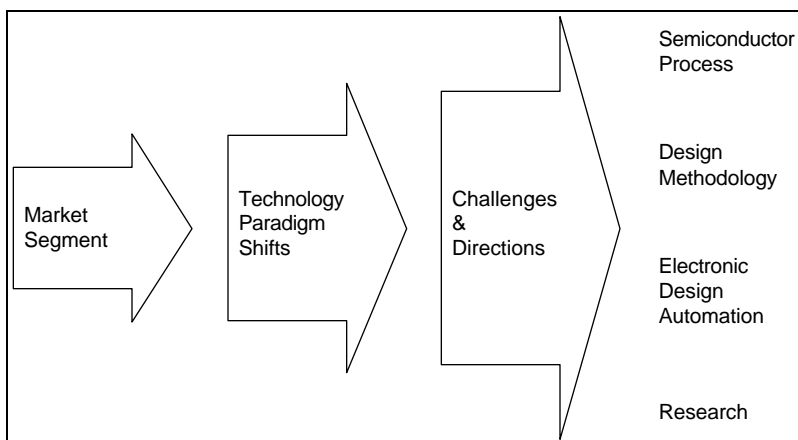


Figure 1: Taskforce Process - Focus on Change

To identify where paradigm shifts will occur, the Taskforce forecasting directions of the semiconductor processing, design approaches, and electronic design automation using the critical design points within these microprocessors.

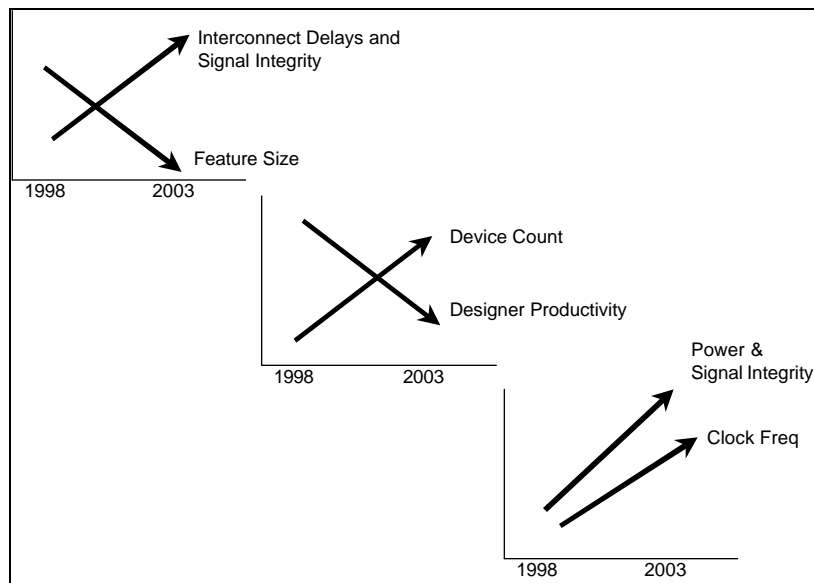


Figure 2: Technology Trends

The semiconductor process information is extracted from the NTRS97. The forecasts presumed that each technology will be enhanced in intervening years. The Taskforce built on these enhancements in order to identify design paradigm shifts and to emphasize other aspects where technology must be reshaped or created.

Next, alternative solutions for the paradigm shifts were examined. The solutions included modification of semiconductor processing, changing the design methodology, and creation of new EDA tools. Some of the technology changes impact design positively.

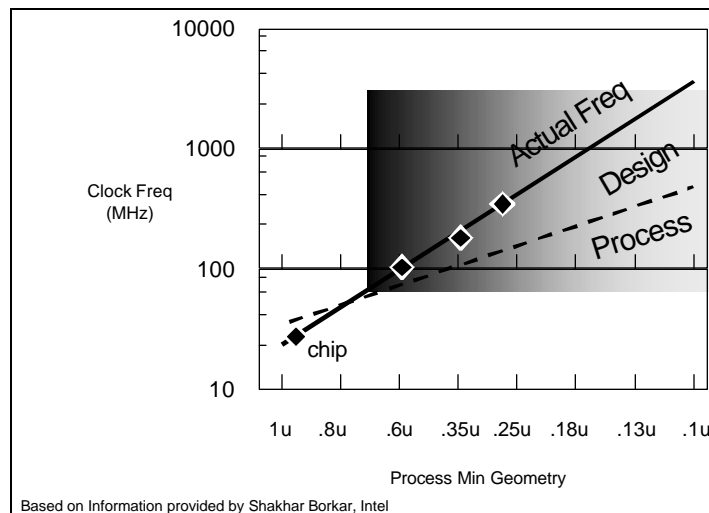


Figure 3: Pushing Frequency through Process and Design Changes

For example, small feature geometries make high-speed circuits possible as a result of the inverse relationship of gate length to RC time constants. But often there are second-order negative effects. For example, increasing the number of transistors operating at high speed, even at reduced supply voltages, will consume greater amounts of power.

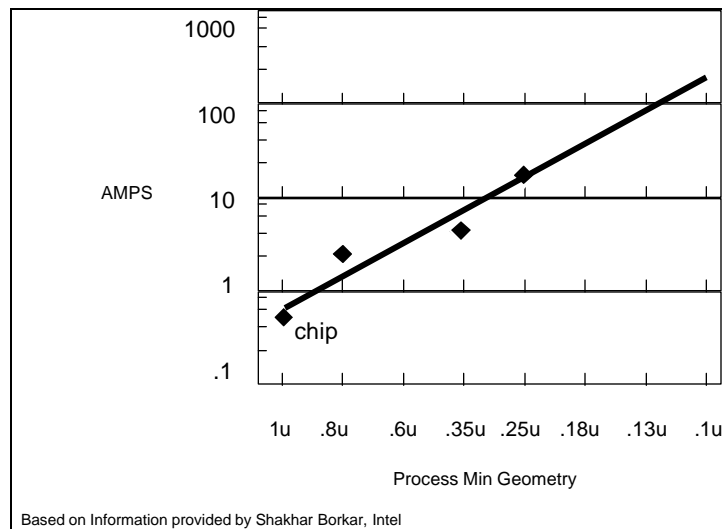


Figure 4: Power Rises Sharply

Target Design

Year 2003 - Microprocessor Data Sheet

The Taskforce felt that it was important to identify characteristics of the typical high-end microprocessor design in the 2003 timeframe to allow evaluation of difficulties and study of possible solutions. These microprocessors will be used in server computers to perform mathematical and data operations, and to perform multimedia and speech analysis / synthesis functions. Chip information, from the NTRS97, for microprocessors that will enter manufacturing in the 2003 timeframe was used as the basis of the evaluation. These products are expected to reach production in the 2004 timeframe. The Taskforce expects that these chips will be comprised of 200 million transistors of which at least 50 million transistors will constitute logic gates. The chip will be fabricated in a CMOS semiconductor process with 100 nanometer or smaller minimum geometry design rules. The overall die size will exceed 500 square millimeters. The chips will be connected into packages using four thousand "bumps", where over half the bumps will be used for power and ground. The chips will run at high speed with basic clock frequencies above 3.5 GHz, and with very fast slew rate requiring internal frequencies above 100 GHz.

▪ Design Size:	
- Total Transistors	200 X 10 ⁶
- Total Logic Transistors	50 X 10⁶
- Wiring levels	8
▪ Scaling:	
- Target Process for Microprocessors Ship)	100 nm (2003 Starts for 2005
- Chip Size	520 mm ²
▪ Frequency:	
- Local Clock Freq.	3.5 GHz
-	3rd Harmonic = 9 GHz
-	Slew rate = 150 Ghz
▪ Chip Statistics	
- Chip I/Os:	4000
- Wiring levels	8
- Total Interconnect length	2840 m/chip

Figure 5: Target Chip Data Sheet

Designing 0.2 billion transistors which will be switching at microwave frequencies will be a severe challenge. It will unavoidably cause several fundamental challenges as a result of the design magnitude, feature size, and clock frequencies.

Design Magnitude: Designing a 0.2 billion transistor chip, while presuming that a typical designer with present automation can design 1000 transistors per day, would take approximately 500 person-years. Even assuming most of the transistors are for memory and the existence of advanced techniques for design reuse, this will be a formidable task. The Taskforce identified designer productivity as a very important area to investigate.

Feature size: With 100 nm being sub-visible light (1000 Angstroms), process assumptions must change, and many current design assumptions must change as well. Design must be accomplished by exploiting a series of simplifications. Design factors previously considered second-order effects will become dominant effects, and it is likely that key simplifying assumptions will no longer be valid. The Taskforce identified the consideration of small geometry effects as one of the changes that needs to be investigated.

High frequency: The microprocessor clock frequency of 3.5 GHz is in the microwave range. If typical signals have sharp edges then at least three odd harmonics will be required to maintain the waveform shape. From a Fourier transform standpoint, the third harmonic is 10.5 GHz, and the slew rates are faster. Design teams must be prepared to deal with harmonic frequencies as high as 100 GHz. Further, at these frequencies, any interconnection approximately one millimeter long will need to be designed through non-traditional (transmission line) means. The Taskforce identified high frequency interconnect as an important area to investigate.

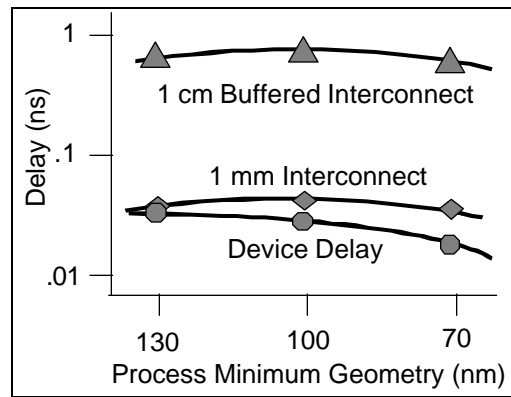


Figure 6: Global and Local Interconnect Delays vs. Gate Delays

The Taskforce objective is to define approaches to design larger chips with fewer engineers, while successfully dealing with new challenges introduced by the sub-100 nm technology. This report identifies enhancements and modifications to semiconductor processing, design methodology, and electronic design automation necessary to reach this objective.

The next sections of this report study these challenges in detail. It then spells out its problem areas and makes conclusive recommendations to deal with these problems.

- Section 2 Power
- Section 3 Signal Integrity and Delay Variation
- Section 4 Design Productivity
- Section 5 Test
- Section 6 EDA System Structure
- Section 7 Conclusions
- Section 8 Further Research Needed

SECTION 2

Power

In the Microprocessor market sector, speed is the ultimate differentiator. Therefore, the designer's highest priority is the maximization of clock frequency for the utmost processor speed. To optimize speed, other constraints need to be relaxed. One design constraint that can be adjusted is power. Microprocessor designers forecasted that, in the future, microprocessors would consume hundreds of watts per chip. Even though there are a number of mechanisms of removing the heat produced from this excessive amount of power, high currents, and thermal gradients are major concerns.

Power dissipation presents a number of design challenges. Inactivating portions of a chip can reduce power, but power-up can cause uncontrolled transients. Few tools are available today to guide designers in analyzing their design for worst-case power. The following discussion emphasizes the electrical effects of power dissipation. However, physical effects also must be predicted and analyzed. A very localized high transient power dissipation can create thermal gradients which initiate cracks in passivation layers and rupture solder bonds. There is a need to develop tools for these analyses.

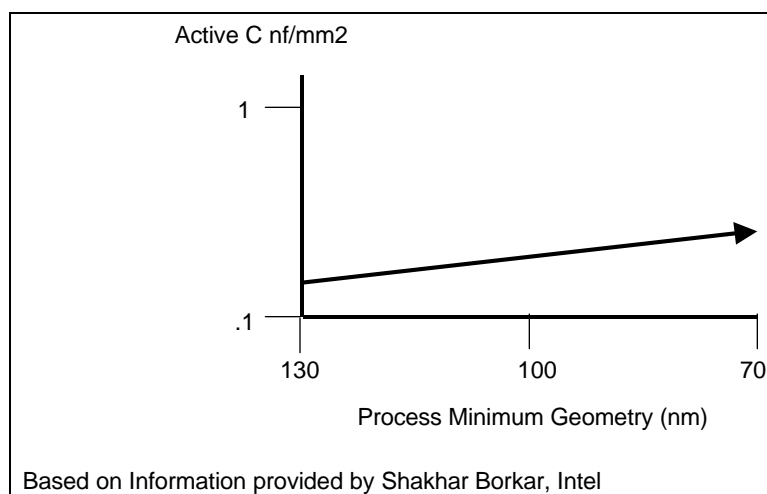


Figure 7: Growth in Active Capacitance Leading to Power Growth

A simplified approach to examine power (P) is to consider it as a function of capacitance (C), frequency (f), and the power supply voltage (V_{dd}). The total capacitance is composed of active and interconnect components that are being charged/discharged between the logic "zero" and "one" levels. As a technology scales down, more capacitance per unit area results, since the gate oxide thickness is decreased and the horizontal wire spacing is reduced (causing increased mutual capacitance between interconnects).

With smaller channel lengths however, transistor drive increases and results in a net increase in the clock frequency (f). Combined with a general trend to larger chips (and therefore yet more capacitance) these two phenomena overcome the gradual reduction in V_{dd} and result in a net increase in power dissipation as technology is scaled.

A rule of thumb for estimating power consumed by a digital circuit is:

$$P = K C f (V_{dd})^2$$

where K is a proportionality constant which accounts for nodes not switching at every clock cycle. To achieve ever-higher performance, circuit families (e.g. dynamic/domino CMOS) are being used that have a higher K constant than classic static CMOS.

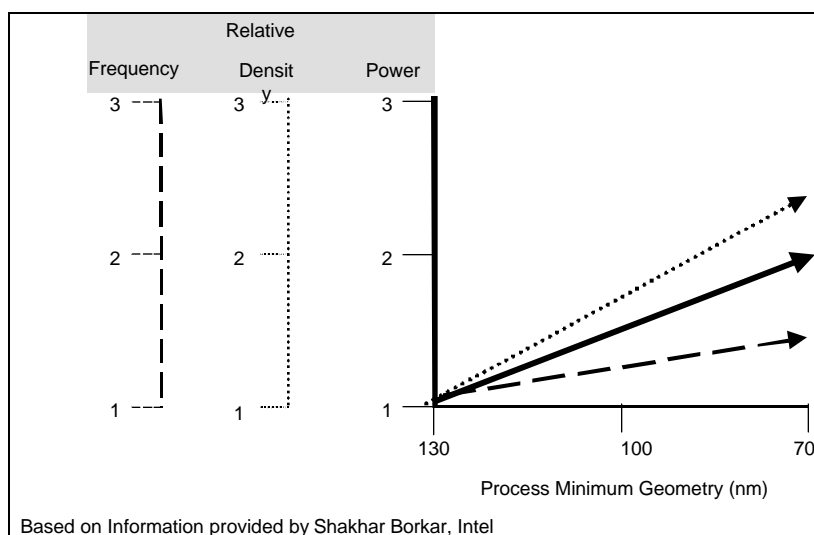


Figure 8: Supply Current

Microprocessors dissipating 100 to 150 watts are currently being designed, and power dissipation of 300 watts are anticipated (predictions of chips requiring 1000 watts were discussed.) This represents a dramatic increase in power from today. Further, since the supply voltage is expected to decline to about 1 volt, the current will increase faster than the power. This means that average currents of greater than 300 amps will be produced, and transient currents will double that value for durations up to 1 nanosecond.

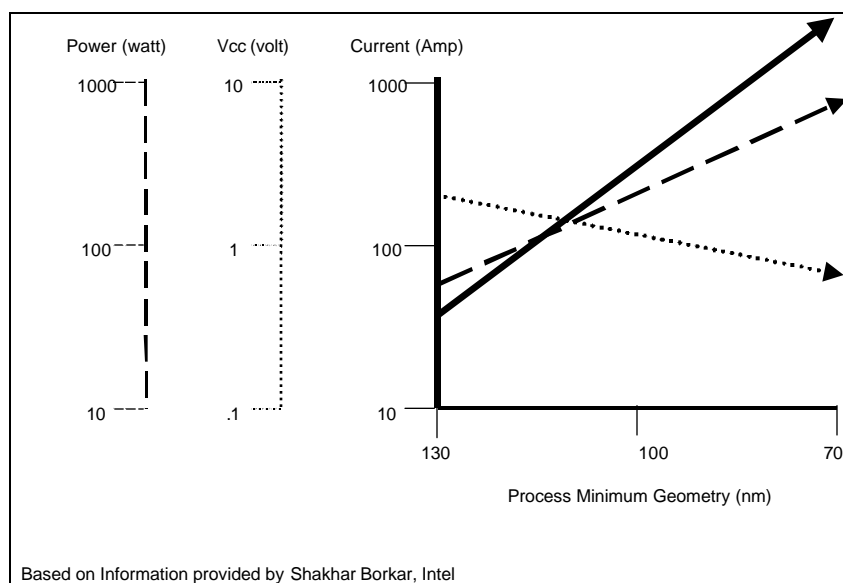


Figure 9: Power / Voltage = Current

A major challenge will be to maintain a near constant supply voltage when delivering (on the average) hundreds of amps, with a variation in supply current of up to hundreds of amps.

With lower supply voltages, the threshold voltage (V_T) of MOSFETs is scaled down in order to maintain performance. Most often this is accomplished by maintaining V_T as a fixed proportion of V_{dd} . Because the sub-threshold current is an exponential function of V_T , this reduction results in a very rapid increase in leakage currents. This leakage current acts to increase the background power dissipation and noise, and is very important for low power design.

With higher power supply currents and increased frequencies, and the use of low-power design techniques that involve clock gating and/or selective shutdown, the current change (di/dt) term increases, while the resistance in the power network is reduced in order to keep rail droop under control. This effectively lowers the threshold at which the dynamic voltage drop ($L di/dt$) is comparable to the static drop (R). This means that rail inductance becomes more important for power supply and noise performance. Even small mutual inductances between the power grid and signal interconnects will disrupt signals because of the large di/dt .

Since power is not expended uniformly throughout the chip, different areas of the chip will experience different temperatures, resulting in thermal gradients across the chip. These gradients may cause timing variability (hot circuits are slower), which in turn can require lower overall design performance in order to compensate for increased timing variability.

There is significant interaction between the power design, clock design, and other global controls. For most microprocessors, the clock network is the largest single consumer of power. The clock runs at full speed over the longest wires thus requiring appropriate ground return path design in order to keep inductance in check.

Due to otherwise excessive power consumption, large chips will turn off portions of their circuitry. However, simultaneously awakening large portions of a chip may cause power surges. Thus, power surges can be expected in common situations such as power up and reset. These are all dramatic effects that will require modifications in processing, design methodology, and in design automation.

For both power and ground distribution and for clock/return distribution, the chip package will play a larger role. The Taskforce foresees increased use of flip-chip technology with thousands of small bumps connecting chip to package with many of them devoted to power and clock distribution with the equipotential planes moving off the chip and into the multi-layer package.

For what remains on the chip, the need to keep electrical noise down demands that the clock network be appropriately shielded to isolate this circuitry. The Taskforce recommends that additional layers of metal be used for shielding as well as power and ground planes. In order to reduce power supply droop, it will be necessary to employ on-chip-decoupling capacitance.

Early in the design cycle, tools are needed to estimate power dissipation accurately for each major section of these very large microprocessors. These early predictions need to estimate power dissipation, thermal behavior, and current requirements for different areas of a chip. Power planning tools must be fully aware of the current distribution and power dissipation of underlying technology. That entails knowledge of wire resistance, capacitance, cross capacitance, mutual-inductance and self-inductance parameters, and layout physical design rules. The prediction must anticipate layout issues such as blockages i.e. portions of the chip where the power grid may be only partial.

In order to reduce the power surges that occur at clock edges, it may be important to adopt self-timed and asynchronous design techniques for major portions of the chip designs. Some designs, which are entirely asynchronous, are already reaching the market (although not at the high end). Current design tools do not adequately synthesize or analyze self-timed and asynchronous designs. The Taskforce suggests that a new category of asynchronous and self-timed design automation tools may need to be invented.

Recommendation I: Power

- **Semiconductor Process Changes Required for Increased Power**

- Additional Metal Layers for Power Planes
- Additional Metal Layers for Shielding
- On Chip Decoupling Capacitors

- **Power Management Design Methodology**

- Increase Usage of Gated Clocks
- Staggered Clock
- Self Timed and Asynchronous Design

- **Design Automation Required for Increased Power**

- Early Prediction of Power
- Self-Inductive and Mutual-Inductive Effects to Signal Line Avoidance Software.
- Power Dependent Timing Verification

SECTION 3

Signal Integrity and Delay Variation

Microprocessors are designed for high-speed operation. The Taskforce investigated factors that limit speeds (GHz) in microprocessors that will be fabricated in CMOS processes with 100 nm minimum geometries. Much of the data that has been used is from the NTRS97 but the data has been enhanced from other sources.

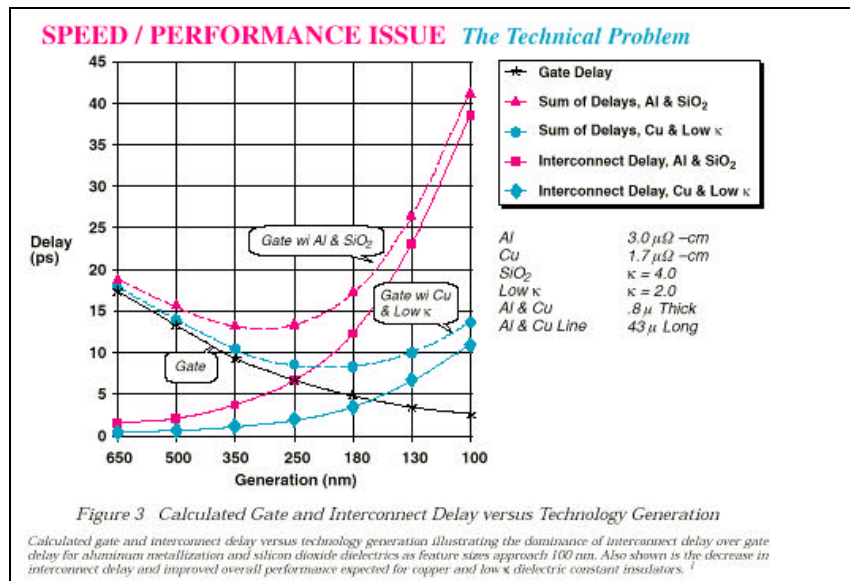


Figure 10: Gate / Interconnect Delay

The gate delay noted in Figure 10 is with no load. The effect of interconnect delay ranges from negligible to dominant. For short distances such as within a cell interconnect delay and signal distortion will be negligible. The Taskforce assumed the use of copper and low-k dielectrics. However, even with the use of copper and low-k materials, the effect of interconnect on delay will dominate over other delay characteristics for major elements on the chip.

Gate Delay	3 ps
On Chip Parameter Variability	+/- 10%
Average Interconnect Delay	12 ps
Time of Flight	5 ps/mm
Interconnect Resistance	100 ohms/mm
Self Inductance Signal Lines	0.5 nh/mm
Mutual Inductance signal to signal	0.3 nh/mm
Crosstalk	0.2 pf/mm
Crosstalk Ratio	.6 $C_{interconnect} / C_{total}$
Reflections	Non-terminated long routes above 9 Ghz
RF antenna	2.5 mm

Figure 11: Signal Integrity and Delay Variation

At GHz frequencies, logic signals will not be clean digital waveforms with slew rates. They will be similar to analog signals that cross switching points. These signals will be distorted through interaction with other

signals. This will upset the signal's integrity and alter the time that the signal will take to reach a switching point. This variable time will manifest itself as variable "delay" time.

Physical phenomena that were heretofore ignored will become predominant. Most notable among these will be self-inductance (particularly in the wide global interconnections), mutual capacitance, and mutual inductance. Mutual capacitance will be noticeable in higher impedance circuits with low current nets, and self and mutual inductance will be dominant in high current nets and especially high di/dt nets. Each will cause the signal's integrity to be diminished.

The mutual capacitance and mutual inductance will couple one signal to another. The couplings (or crosstalk) will be proportional to wire dimensions and pitch, and the relative slew rates of the two signals, so that the amount of coupling will vary based on functional operation. One effect of this may be that signals transition before, during, or after the time they need to switch. Pre-charged networks may present incorrect information to the following stage switches. Waveform glitches may impact delay even more significantly if they occur at the switching point where they can cause an actual change of state.

Long word-length microprocessor architectures with multi-clocks were examined. These synchronous microprocessors switch many nets simultaneously. The advantage of long word architectures is that many parallel operations can be accomplished concurrently. However, this leads to multiple signal integrity problems. In many cases, combinations of mutual capacitors and inductors will affect many signals.

A simple pair of interconnects is used to illustrate the mutual interference problems. One signal is considered of primary interest. The primary signal's response is affected by another signal, which is an aggressor. If a primary signal and aggressor transition close to the same time, then the timing of both signals will be modified from nominal. For example, if both signals transition at the same time with the same polarity, then there is no change in charge across the mutual capacitance between them. The signal will change faster on both the primary and aggressor nets. If the signals are opposite in polarity, then the mutual capacitance will require double the normal charge and will lengthen the delay on both primary and aggressor nets.

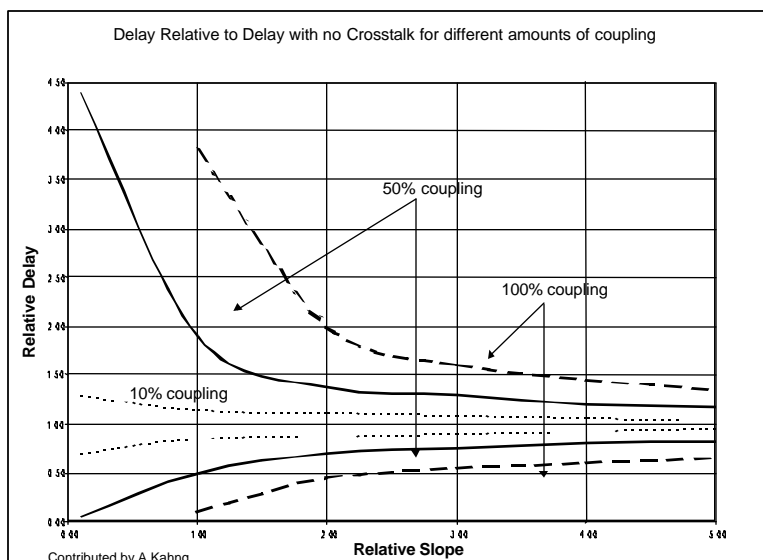


Figure 12: Delay Variation

During multiple signal transitions, some signals will rise faster and some slower. Additionally, the signals will be seriously distorted. Because 100 nm geometry transistors are low gain, some of these distortions will be transmitted to subsequent stages. These stages will have other aggressors that add other distortions. The manifestation of distortion in a digital system is delay variation. Even pre-charging a net will result in delay variation. Each of these distorted signals will reach its final value if given enough time.

However, in high-speed designs, precautions will need to be taken to assure that the variations will not cause a system malfunction.

Interconnections of approximately 1000 nm (10x minimum geometry) or less may remain unaffected by other signals. For interconnections that are approximately 100u in length (1000x minimum geometry), the effects of other signals will limit performance and precautions will be needed. Long interconnects at the chip level, like buses, clocks, reset, and power distribution, must be designed to control the effects of mutual signal distortion. Tools may be needed to support several classes of interconnection resources, depending on length and delay/noise criticality. In an interconnect-centric world, the available types of interconnections between blocks become a critical class of design options.

At some point, buffers will be added to reach a practical maximum delay per mm. However, when using buffers/inverters, time of flight increases. Inverter insertion will be most effective if inserted in a physically staggered pattern. The staggering will have the effect of balancing the aggressor signals by inverting the aggressor's polarity. This will limit the effect in some challenging situations, while it does not help in other situations like power distribution and low skew clocks. The primary benefits of adding buffering are the reshaping of the signal and reducing the effects of cross capacitance. The use of Schmidt Triggers or similar circuits with hysteresis may be more effective in minimizing the impact of delay variations.

Designers will also need to consider the use of shielding to limit the effects of long parallel interconnections or high current. Running signals over a functional block will often induce undesirable effects on the block's timing. Shielding will be needed to greatly reduce signal interference, but will need to be carefully designed in order not to introduce undesired side effects.

Delay uncertainty will be a function of the direct and indirect relationships between signals. If multiple signals are always spaced temporally, then their mutual effect will be diminished. If these signals often switch at the same time and at random polarities then many will impair the timing of the others.

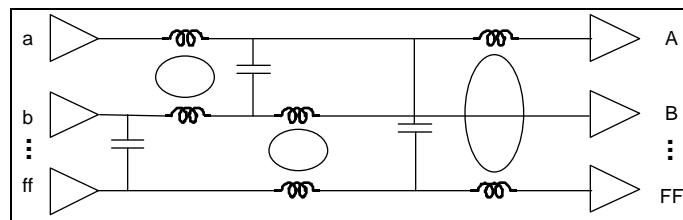


Figure 13: New Signal Interrelationships

Microprocessors are synchronous and many signals transition concurrently. To avert problems, designers and / or tools need to look for long runs of parallel wires and analyze them for coupling problems.

There are several essential EDA technologies. One such technology, parasitic extraction, derives the mutual inductance and capacitance as well as other parameters that affect delay. Because signal integrity analysis must capture relationships between the signals, delay models will be multi-port and statistical. Additionally, parameter and temperature variations across a chip will also contribute significantly to changes in delays. Therefore, additional ports will be needed for these variables. These models will be evaluated in a multi-path logic analysis engine.

The Taskforce recommends that the multi-path logic analysis engine be built into physical design. This multi-path analyzer will need to carry the relationships between signals so that mutual relations will be properly modeled. Physical design must be "signal integrity" aware. Tools must support transparent maintenance of timing models and avoid timing problems whenever possible by using buffering, shielding, variable and asymmetric width / spacing of interconnections, and interlacing quiet signal lines next to active signal lines. The higher frequencies will require the equivalent of twisted pairs, coax cable, transmission lines, and even wave-guides. Sensitive circuits, e.g. those which cannot tolerate the added capacitance and/or inductance of parallel shielding lines, may need to employ driven shields using techniques from analog circuit design. EDA tool enhancements will be needed to include such circuitry.

The Taskforce recommends that physical design be “interconnect-centric”. The Taskforce further recommends that physical design be completed in hierarchical stages and that longer, more challenging connections be analyzed first, and then either completed or redesigned before dealing with less challenging details. The higher levels of physical hierarchy with power delivery, global signals such as clocks, and busses and signals between blocks will be designed first. During the design of the chip level, the lower level functions will be specified and budgets will be set for functional delays. Functional design at the high level will control signal integrity by minimizing the number of signal transitions that implicitly interfere, and by minimizing the number of global signals that are required. Thereby, elements at lower levels of the design hierarchy will be designed aware of chip level signals that will dominate delay. This lower level of physical design will concentrate on achieving the intended delay budgets.

Recommendation II: Signal Integrity and Delay Uncertainty

- **Semiconductor Process Changes**
 - Additional Metal Layers for Shielding
 - Low mutual capacitance and low mutual inductance between signals including power
- **Design Methodology**
 - Hierarchical Design that is Interconnect-centric
 - Staggered Signals
- **Design Automation**
 - Physical Design that is Signal Integrity Aware
 - Multi-Port Delay Models
 - Multi-Path Timing Analyzer
 - Interconnect-centric Design Tools which emphasize High Level Physical Design

SECTION 4

Design Productivity

A major Taskforce objective was to find approaches to designing microprocessors using fewer engineers and with shorter cycle times, while being concerned with more complex electrical/physical details.

Details dominate the effort to design a chip. Many of the simplifications applied to the chip design process today will no longer be valid. The good news is that designers are a resilient and resourceful bunch, and will keep designing high-end microprocessor chips in spite of the roadblocks thrown in their path. However, billions of transistors, switching at microwave frequencies, and complex cross-signal interactions will exceed any engineer's ability to analyze the entire chip design and uncover timing problems. In previous generations, designs were limited in speed by the transistors, and interconnect was usually considered contributing second order effects. In future generations, speed will be limited by interconnect and transistor performance becomes a second order consideration. Solving interconnect difficulties will dominate design cycle. The design system and methodology must evolve to support such changes.

Because most of the timing effects will be layout related, simple functional analysis will not be effective for physical design verifications. Current analysis techniques such as logic and circuit simulation have already exceeded their limits. Static analysis that is based on gate or transistor verification will overwhelm the designer with exceeding large numbers of false-path errors.

However, the time it takes to deliver a good chip design depends on many other critical issues such as, a wrong or incomplete specification, or poor design practices. Only by attacking all critical issues, will improved design productivity result.

PRODUCTIVITY FACTORS

Design productivity is a by-product of two interrelated activities. The first is the efficiency of the team producing the design. Microprocessor design teams of today have a wide variation in the team sizes. Some teams are as much as an order of magnitude larger than others. Factoring in design differences do not account for this wide variance. The factor that appears common to the most efficient design teams is that they are populated with "tall-thin" designers who had previously worked together as a team.

The second is the number of design iterations (or spins) required to achieve a correct implementation. The most effective way to shorten design cycle time is to reduce design iterations. One of the major causes of multiple iterations is inconsistent views of the design. Often, one team does not know about the latest changes made by another team or individual. Similarly, because the original specification may not be clear, one team may think that a bus address is 0:32 while the another thinks that it is 32:0. Often a designer will make a change to the design in his personal design responsibility scope and verify it, but forget to update the master design database.

EDA systems must assist both of these interrelated activities and designers' capability must grow so that they are capable of working across hierarchy. EDA needs to optimize design processes and focus design activities and methodology so as to allow the engineers' efforts to be focused on the most critical tasks.

GUIDELINES

A set of guiding principles is offered for creating High-end Microprocessors using 100 nm processes. These guiding principles may seem simple but their impact can be enormous.

- Avoid problems rather than identifying and correcting the errors later

Verification is the biggest bottleneck in the design process. Tools must be provided which do not require repetitive verification. Development of verification management methodologies, and supporting tools, is required so that automation simplifies the verification process. For example, commonly, a cell is designed and then placed and routed. This is followed by an

analysis of the interactions between that cell with the surrounding interconnections and other circuitry. This analysis may then uncover a resulting design error, which causes the design loop to be reentered. Design teams can prevent problems by defining and following strict design rules. A typical rule may be to include buffering and shielding during the design of a cell to make it resistant to signal interference.

- Verify once and use the results forever

Designers are encouraged to thoroughly evaluate each hierarchy level and encapsulate that design entity as a known-good design. The encapsulation will be used within the next higher design level. Each encapsulation must encompass a broad range of design parameters. Certainly among these parameters are function, timing, signal integrity factors, power, instantaneous current, size, pinout, and design assumptions or constraints.

- Change design focus to be interconnect-centric

The timing of 100 nm designs depends upon the interaction between signals. Characteristics of all but local interconnections dominate over active devices. Physical and electrical design must consider interconnect effects first, then the active devices. What is suggesting is to forecast the interconnection properties then later, precisely define interconnect details. Use the forecasts and a precise layout to refine the acceptable range of each property throughout the hierarchy.

- Tether design changes to the old versions to control the magnitude of any change

The Taskforce suggests that controlling the effect of a change is as important as the change itself. The tools must track changes and their hierarchical effect so that there are no surprises during implementation.

Recommendation III: Guiding Principles

- | |
|--|
| <ul style="list-style-type: none">▪ Avoid problems▪ Verify once▪ Interconnect-centric design▪ Tether design changes |
|--|

In the light of the preceding guiding principles, several strawman design approaches were evaluated. Some were not capable of achieving high-speed 100 nm designs. Others would be burdened with immense details at high levels of hierarchy. The Taskforce feels that the following design approach will meet the challenges of 100 nm microprocessor designs.

MEET-IN-THE-MIDDLE

It is felt that a formalized meet-in-the-middle design approach will be extremely advantageous. Design will be hierarchical. As usual, the design process begins with architectural definition and ends with a physical realization and testing method. The intersection of these domains is the middle. For microprocessor design, the middle is the full chip level. The full chip level is where the lowest level of architecture design and the highest level of the physical design meet. In the middle, the process of resolving ambiguities begins by recognizing risks and clearing away conflicting requirements.

Recommendation IV: Meet in the Middle Design Approach

- **Semiconductor Process Changes**
 - N/A
- **Design Methodology**
 - Actively Avoid Problems
 - Add Shielding*
 - Add Buffers*
 - Interconnect Models*
 - Verify Block*
 - Full Chip Level First
 - Full Chip Layout*
 - Forecast Block Specifications*
- **Design Automation**
 - Verify Chip Using Models

AVOID PROBLEMS

Applying the four principles (see Recommendation III) leads to some conclusions. Avoiding problems implies that bottom-up design cannot be performed without thoroughly specified goals. Goals need to encompass the full spectrum of design parameters from delay to signal integrity, and from power/current requirements to area/pinout. The goals must be based on credible estimates and be as accurate as practical. Timing information, which heretofore was a back-annotated afterthought, needs to be accurately predicted and budgeted. For critical parameters, the degree of uncertainty must be forecasted.

ESTIMATION

Budgets are a means of passing constraints and ranges of ambiguity down hierarchical levels. The budgets must, at each hierarchical level, enable realizable designs. One characteristic of efficient design teams is the ability to estimate/forecast well. These teams have learned to estimate efficiently because they have experienced tall-thin designers who can focus effectively across design levels. Their high level decisions are a natural outcome of low level consideration. It is suggested that estimation become institutionalized and embedded within the EDA infrastructure. Special methodologies and EDA tools need to be developed to estimate the key properties of each block in the design hierarchy as accurately as possible.

Estimation is foreseen as a joining of top-down specification and bottom-up prediction. Prediction is never exact, but the quality of the design in the middle depends on being close. For the full chip level, it must be close enough to complete the level designed. The design will include power distribution, built -in test capabilities, dominant signal (such as clock and reset) distribution, inter-block timing, and soft error control.

Estimating is a conjunction of historical information and tools. Nearly every microprocessor design team has individuals who have designed microprocessors previously. Once captured, the historical experience can be used as a starting point for the next microprocessor design. The same is true for the large blocks (intellectual property). It is expected that automatic generators will create data paths, register stacks, and memory (these generators can also build the models.) It is expected that synthesizers will create random logic. These synthesizers will be advanced in order to produce increasingly accurate models for high-level verification. Designers, at times, may resort to prototype experiments using new blocks or models. However, model building should be automated with special tools that confirm accuracy of the generated models.

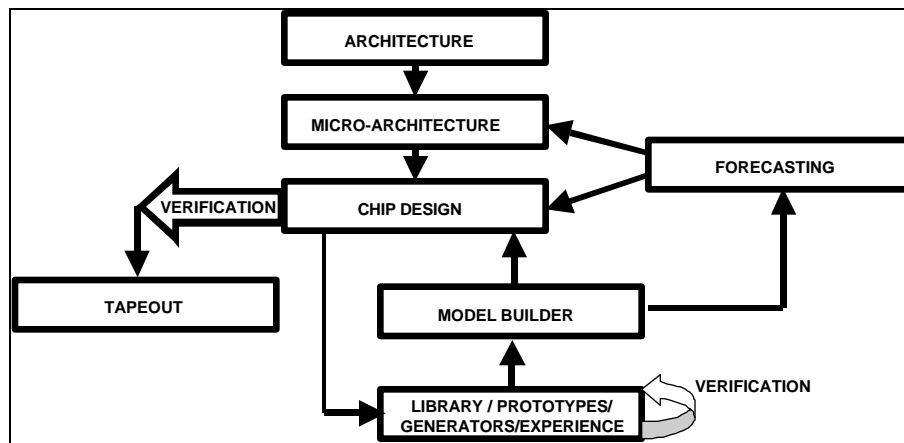


Figure 14: Interconnect-centric Design System

Some design methodology changes will improve the quality of estimation and of the chip design - whereas, software can prevent problems by active avoidance. One example is adding shielding between interconnects that are likely to interfere. This shielding stabilizes the circuit operation and can minimize the effect of an interconnect on surrounding entities. It also will make it possible to predict a block's operation more accurately. Problem avoidance reaps a side benefit of predictability and although such avoidance methodologies may appear at first to require additional silicon area, correct operation forces that result. It is suggested that these schemas be increasingly leveraged.

SIMPLIFY THE DESIGN PROCESS

Some methodology changes can simplify the entire design task. Power and signal integrity are both affected by the number of signal transitions. Minimizing the transition counts and (especially) controlling transitions on longer length interconnects will reduce power and noise problems.

With meet-in-the-middle design techniques physical design starts at the full chip level. As was stated earlier, design needs to be interconnect-centric, since interconnect has the largest effect on design with longer interconnections having the most effect. This includes all of the long signal lines, busses, global clock signals, reset signals, test signals, and power distribution lines. The longest interconnections are at the chip level, then at the block level interconnect, and finally at the local silicon and cell level. To design these interconnections, the high-level blocks of a design must be estimated. The blocks must be placed in the chip structure and interconnects between them will dominate the performance. These blocks are thereby specified in-place where they will be used. Care can be taken to reuse blocks repetitively to save design and verification time.

After global chip level interconnect is completed, each higher level block specification is enhanced with the dominant interconnect effects. These include loading effects, signal integrity effects due to inter-layer crossings or nearby interconnect on the same layer, clock/sequence skew, and power fluctuations. The block's pinout and dimensions are also delineated. This enhanced block specification allows a team to have, at all times, an intact hierarchy that is divisible into sub-activities that can be reassembled. Since each block always coexists with its environment, the design is successively refined and a modification within one block can be reflected to the rest of the design through the chip level hierarchy. Thus, the new design is tethered to the old design and degree of change can be controlled.

Another advantage of meet-in-the-middle is early verification. The estimated blocks are complete models, where lower level details (behavior, delays, signal integrity, power/current, area, and pinout) are abstracted into high-level block representations. Often the high-level blocks will have built-in checkers to assure that their functions are being used appropriately. As a design proceeds from estimation to realization, the model/verification accuracy will improve. As a design matures, the verification is completed hierarchically. As blocks are refined, only the modified blocks will need new parameterization and verification will be maintained. To make this possible, the details of the each design entity must be

captured using physical data (layout) whenever available to increase its accuracy. In this way, engineers can concentrate on the interconnect between the modeled functions (entities).

Controlling change is also important to assure convergence and to limit the amount of (re)verification. Whenever a design modification is proposed, it must be tethered to the preceding design by determining that it meets block and chip level constraints.

The process is one of successive design refinement cycles, converging to a well-formed complete block design. During the refinement process, advanced tool capabilities should transparently (at least until a design error is detected) maintain models defining block properties, and verify that successive refinements are converging upon a well-formed result. Some tools may also need to accomplish trend analysis and be capable of issuing warnings, providing diagnostics, and escalating issues.

MODELING

The lower levels of detail are abstracted into models that are used at different levels of the hierarchy, up to the blocks at the full chip level. This is reverse synthesis. Handling detail at a high level requires elaborate modeling techniques. These techniques trade the difficulty of analyzing a billion individual elements for the difficulty of building abstracted models. Verification is an NP complete problem, but model building time and accuracy can be traded off against one another to minimize the effort. For example, power can be modeled as turn-on current, reset current, clock edge current, nominal operation current, worst-case operating current as well as final-test current. Each element of the model requires an evaluation and these evaluations must be automated so any entity can be rapidly evaluated so that the higher levels of modeling can utilize the lower level parameters

In addition to the four guiding principles of Recommendation III, some additional guidelines are recommended:

Recommendation V: Additional Guidelines

- **Semiconductor Process Changes**
 - N/A
- **Design Methodology**
 - Reduce design verification by making designs regular.
 - Simplify design by adding microcode to reduce hardware complexity.
 - Plan power, signals, soft error handling, testing approaches, area and pinout at the chip level to make creating detail immensely simpler.
- **Design Automation**
 - N/A

FORECASTING

Design methodology must change to enable 100 nm microprocessor design. New design automation software is needed to support this new methodology. A primary requirement is an enhanced ability to forecast and estimate block characteristics before they exist. Another requirement is for compilers that build models that the forecaster and high-level verification can use.

- **Budgets / Specifications**
 - Power Distribution
 - Built -in-Test
 - Dominant Signal (Such As Clock and Reset)
 - Block Delay Distribution
 - Signal Integrity
 - Soft Error Control
 - Area / Pinout
 - Function
- **Audit Design vs. Budget**

Figure 15: Forecasting

Forecasting is intelligent estimation based upon libraries and captured knowledge. It helps model early stages of a design and audit the later stages of that design. The objective is to continuously build a reservoir of design knowledge and detect design problems by analyzing differences within the information. The estimation process functions by extrapolating from current available data to a design proposal. When the extrapolation exceeds limits, more data is requested. This data may be supplied through experience, libraries, generators, or prototyping. As a design matures, the Forecaster audits implementations to assure that specifications are met. It interrogates a wide range of characteristics such as soft errors, signal integrity, power, testing, and area compliance. If a design exceeds any bound, the designer needs to be modified as early as possible. The approach must allow for successive refinement of a design and build a multi-team concurrent environment. For example, if the chip current exceeds the power capacity, then either the power rails must be redesigned or the chip must be re-architected to reduce power requirements.

- **Estimation Basis**
 - Experience
 - History
 - Intellectual Property
 - Generators
 - Prototyping
- **Model Building**
 - In-place Models Including Interconnection
 - Backannotate Physical Design Characteristics
 - Full Range of Design Parameters

Figure 16: Automated Model Builder

The Forecaster is a key element in an interconnect-centric design environment where interconnect is designed first with estimates of the functional blocks. This code must extract signal interference, block timing uncertainties, and power consumption from the physical design.

Another key element is to allow estimation and knowledge collection. The design automation process that is proposed requires that many models be generated rapidly. Therefore, an Automated Model Builder is paramount for a productive design environment. Without this capability designers may be forced to trade-off design efficiency against modeling expense. This would lead to no net gain in productivity. The Automated Model Builder must absorb the physical design characteristics for all the levels of design including the chip. It abstracts lower hierarchical level information in a form that higher levels in the hierarchy can use. This extraction must cover the full range of design parameters including signal

interference, block timing with uncertainties, and power consumption. Further, the block level model needs to support enhanced delay variability due to the uncertainties of circuits and logic paths.

The Taskforce proposes significant changes to design methodology and design automation. These changes are directly forced by 100 nm processing. These changes are also forced by microprocessor design's continuance to grow in complexity, which magnifies the need for productivity. The design methodology proposed is built on the principles of problem avoidance, verification only once, interconnect-centric design, and the tethering of design changes. The Taskforce recommends that efforts focus on these new methodologies and design automation as early as possible so industry will be ready for the challenges ahead.

Recommendation VI: Productivity

- **Semiconductor Process Changes**
 - N/A
- **Design Methodology**
 - Meet in the Middle Design Approaches
 - Guidelines
 - Verify Using Hierarchical Models
- **Design Automation**
 - Forecaster
 - Auditor
 - Model Builder

SECTION 5

Types of Testing

Testing is done many times in the design and manufacture of Integrated Circuits. Design engineers test to confirm their design and to characterize it. Manufacturing tests to reject bad devices, and in sample quantities to confirm reliability and maintain processes within specifications. Where incipient early failures cannot be detected in final test, devices must be burnt-in and then re-tested, a costly process step to be avoided if at all possible. Each will have its problems in 2003, but because the manufacturing tests contribute by far the largest set of problems, the Taskforce concentrated on them only. Manufacturing's screening tests are not only costly in themselves, but are costly in yields when they err on either side – rejecting good devices or allowing bad devices to continue processing or be shipped.

Electrical Testing Challenges

Designs are moving ahead at a faster rate than current tester technology. The amount of circuitry is exploding, requiring huge numbers of test vectors. Architecture is increasingly complex – many diverse functions on the same chip demanding a great variety of test methods.

With chip clock frequencies moving to 3.5 GHz, and with edge-rate frequencies moving well beyond that, target testing frequencies and complexity of test head circuitry become severe problems. Variation in interconnect parasitics (complex distributed R-L-C-M due to coupling and over-the-cell routing) can cause timing variations particularly difficult to test since those effects are often signal-combination-specific minor changes in timing and signal levels. This often manifests itself as clock jitter.

Some blocks on the chip may have been acquired as Intellectual Property (IP) from outside sources, and the details of their design not known even by the design team itself. The tests required by these blocks may not be known in sufficient detail.

Physical and Thermal Challenges

Making contact to chips with 4000 I/O bumps switching hundreds of amps of current, some with power sources less than 1 volt will present real challenges at the wafer probe stage. Electrically, high currents must flow through very small pressure contacts with tolerable voltage drop and with acceptably low noise. Thermally, while packaged devices at final test can be clamped into test fixtures approximating their final assembled configuration, this is not possible at wafer probe. As chip sizes increase, the contact heights become inherently less planar, increasing the problems of making solid, reliable temporary pressure contacts to them.

Heat cannot flow through the pressure contacts in the same way that it can through bumps when bonded into packages – means must be found to provide reliable, adequate heat flow and to reduce heat generation in the wafer probe stage.

Traditional test methods cannot continue to be employed which depend upon accessing internal nodes. Geometries become ultra-fine, multi-level interconnects make much circuitry inaccessible, and when packaged by flip-chip techniques, back-access is nearly prohibited.

Some proposed test methods, such as using Electron-Beam technology, impose environmental requirements such as vacuum, which will only compound these problems.

Economic Challenges

With feature size rapidly reducing and with wafer and die sizes remaining the same, manufacturing cost per transistor is declining rapidly. Unfortunately, if traditional approaches to manufacturing test continue to be

pursued, equipment will continue to become more complex and therefore more costly, and testing times will lengthen, requiring more of the expensive machines. Test equipment depreciation per transistor is forecasted to at best remain flat. Figure 17 shows these two relationships both historically and as forecast by the NTRS97 and other industry information.

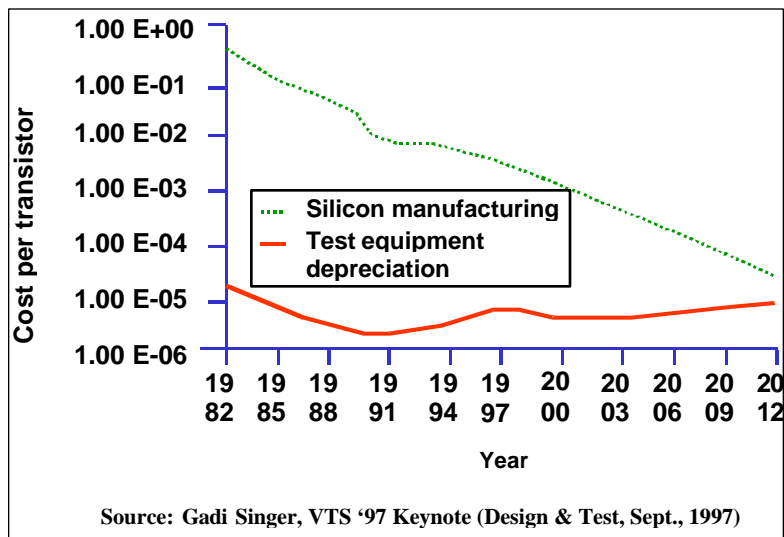


Figure 17: Test Cost Impact on Product Pricing

Test Yield Challenges

Product requirements and feature size scaling will result in silicon speed increases that will increase faster than the required increase in overall timing accuracy (OTA) of traditional manufacturing test equipment.

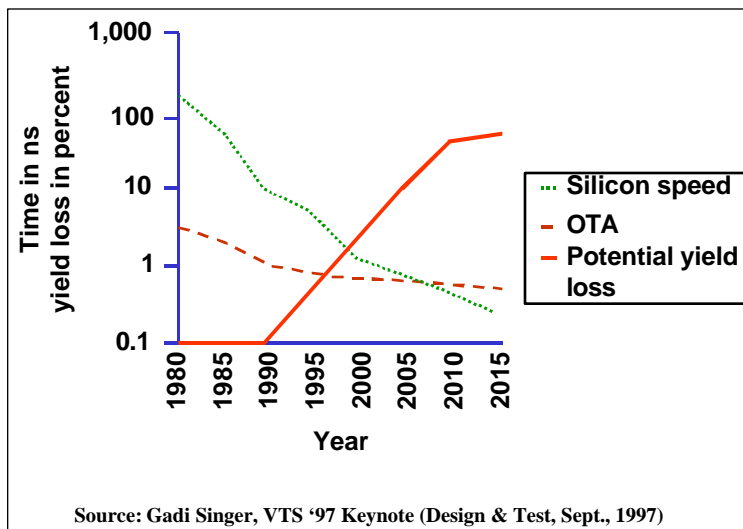


Figure 18: Yield Loss due to Guard Banding

Figure 18 shows these two relationships both historically and as forecast by the NTRS97 and other industry sources. Manufacturing test use of increased guard-banding against timing defects to compensate for the loss of timing accuracy would result in a finite and significant manufacturing YIELD LOSS. The Taskforce feels that this would unacceptably increase unit costs, so that increased guard-banding cannot be used, and other solutions must be found.

Burn-In and Reliability Challenges

With the increased number of chip input/output pins, increased supply-current, increased power dissipation, and rising chip-to-substrate interconnection complexity, the cost per burn-in socket is expected to rise sharply. In addition, alternatives to uncover potential chip manufacturing failures, e.g., use of I_{ddq} testing, are at risk for use in the future. The figure 5.3 shows these two relationships both historically and as forecast by the NTRS97 and other industry information. Alternatives to burn-in must be developed.

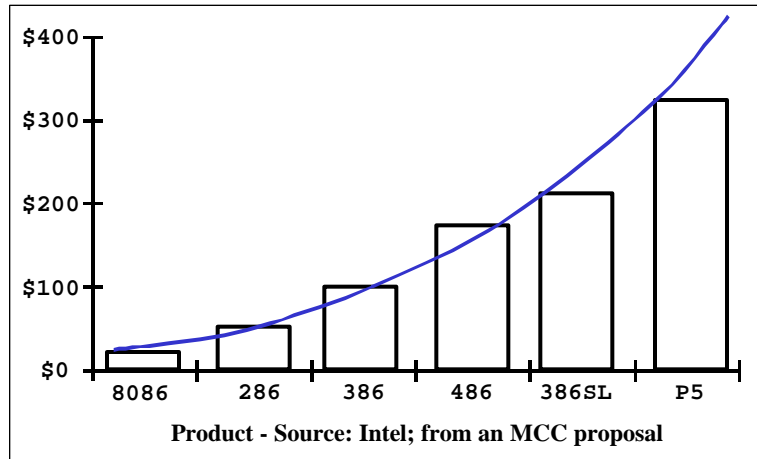
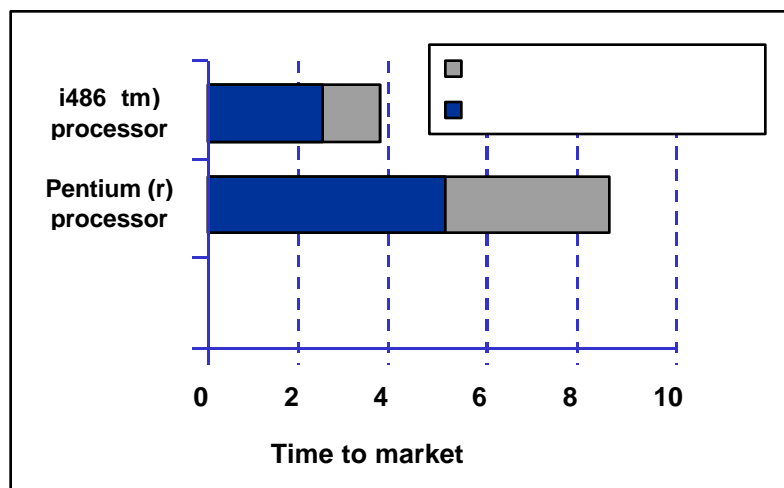


Figure 19: Cost Per Burn-in Socket Position

But the early failures against which burn-in is targeted are not the only reliability consideration. Some fabrication processes are being dramatically changed as we move to 100 nm devices, particularly the replacement of aluminum with copper in interconnect layers.



Source: Carbine and Feltman (Intel) at ITC

Figure 20: Time to Market and Volume Increasing

These dramatically change the long-term reliability picture. Failure models since the invention of the integrated circuit have been built around the properties of aluminum—electromigration in particular. Many design “rules of thumb” have these models at their base. Just as designers must not continue using obsolete rules, test engineers must not continue using reliability projections and make reliability tests based on incorrect models. A new set of fault models is needed.

Time-To-Volume Challenges

Time-to-Volume is as important as initial product release and overall Time-to-Market. This is especially true for products targeted at low end, high volume markets. The below figure shows these two relationships both historically and as forecast by the NTRS97 and other industry information. Increased test equipment complexity and therefore procurement time could delay manufacturing ramp-up unacceptably in 2003 – other solutions are imperative.

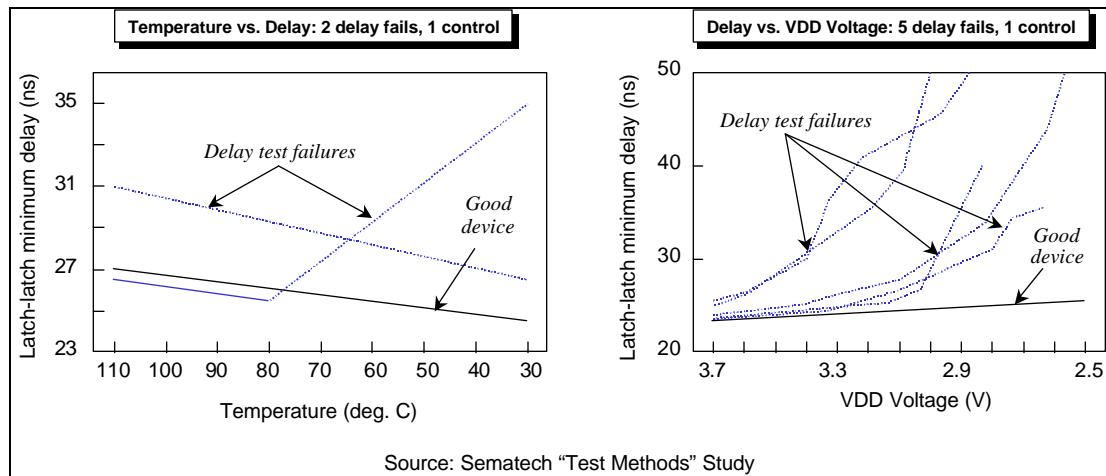


Figure 21: Temperature and Voltage Stress Identifies Faulty Behavior

Defect Identification through Accelerated Stress Testing

An important 100 nm test challenge to resolve will be to establish if parametric stress conditions can help to distinguish defect behavior from normal behavior, for example, the use of temperature stress or voltage stress in finding defective devices. It was noted above that I_{ddq} may no longer be effective, yet elimination of burn-in is crucial. Figure 21 shows example data illustrating the possible roles of parametric stress testing in manufacturing testing.

SHIFT TO BUILT-IN-SELF-TEST (BIST)

The Taskforce recommends full implementation of BIST (Built-In-Self-Test - the chip should test itself.) The challenges above will only be met by taking a fundamentally different approach to manufacturing test. The function of the test hardware should be reduced to providing power, to triggering test initiation, and to collecting pass/fail data from the chip itself as it performs its own tests.

Fortunately, high-end microprocessors are uniquely suited to implement BIST. Inherently on the chip are the controls, clocks, memories, registers, timers, and level-differentiating circuitry needed to implement full-chip BIST. Some increase in these facilities may be required, but its cost is miniscule compared to the alternatives discussed above. A side benefit of BIST is that it remains available for use throughout the life of the microprocessor, and can be made a part of the end system's self test as memory test is used today.

Using BIST eliminates all the timing accuracy problems, the contact parasitic effects and the test head complexities. A timing test can be performed totally on-chip, by initiating a sequence of actions through a significantly long path and looping the result back to a timed gate – if the signal arrives within the time window that the gate is clocked open, the device passes, otherwise a failure is registered. We are not aware of any specification which could not be tested using fully on-chip BIST.

The reduction in complexity and therefore in cost of test equipment would be dramatic. Test time itself could become less important, since many chips could be testing themselves simultaneously. Only power supply and heat removal problems would need consideration.

Implementing BIST requires that it be an integral part of the chip design, hence our next recommendation:

DESIGN TESTS DURING CHIP DESIGN

Design for Test (DFT), if it is to be based on BIST, must be carried on concurrently with the R&D effort to design the chip itself and at as high a priority. No longer will it work to fully design a chip to meet its specifications and only then turn it over to Test Engineers to figure out how to test it.

No fundamentally different capabilities or architectures will need to be designed into the chip to implement BIST, but the required number, location and properties of the various elements must be included. Hence, Design Reviews should also be Test Reviews, and Test Engineers be an integral part of the design team.

A new set of CAD software will be needed to support this function. It should monitor coverage, reduce the "bookkeeping", insert the tests themselves and generally be an integral part of the design flow, interfaced well and seamlessly into that flow.

DEVELOP NEW FAULT MODELS & PLAN TESTS TO THEM

Manufacturing tests are performed solely to detect faults. Every source of potential fault must be identified, its probability of occurrence and severity of effect known and then a test strategy developed and implemented to detect it. At the root of this effort is the fault model.

New, reliable fault models will need to be developed for all of the types of net and signal interference, including complex R-L-C-M parasitic coupling and fault modes for both over and through the cell routing. The longer-term mechanisms of failure must be re-evaluated. Where they have changed, for example through the use of copper instead of aluminum, changes in screening, sampling and environmental testing must be implemented.

Universities have taken the lead in the past in much of this modeling work. This is a fertile area for them to contribute again.

Recommendation VII: Test

- **Semiconductor Process Changes**

- N/A

- **Design Methodology**

- Design Tests during Chip Design
 - Test Engineer must be integral member of design team*
 - New CAD software needed for DFT*

- **Design Automation**

- Develop New Fault Models & Plan Tests to Them
- Copper and other processing changes will invalidate some current models
 - Effective testing strategies must be based on good fault models*
 - Fertile area for University participation*

SECTION 6

EDA System Structure

As the number of tools and the kinds of data that have to be passed between them increase, these tools must become reusable software components that make their data and services available to other tools via standardized Application Procedural Interfaces (APIs). Two facts are coming together that make this important. The first is that the number of tools will increase along with the growing number of design issues that must be considered. Thus design flows will require the use of more tools and, coupled with a growing level of design abstraction, there will be steady increase in the necessity for shared data between tools. Therefore, design systems will need to be more than mere collections of tools, but rather tightly integrated suites of tools that explicitly support homogeneous design flows.

Data needs to be managed consistently throughout design flows, without forcing an explosion in the number of file formats or undue redundancy in persistent data. Net, Pin, Cell and Instance names tend to be passed from the top of the design hierarchy to the bottom, passing through all tools. Parasitic information is passed from the extractors to timing analyzers and simulators, bottom-up. Constraints are propagated and enhanced in detail as they pass from floor-planner to synthesis tools, to placers, to global routers, and on to detail routers.

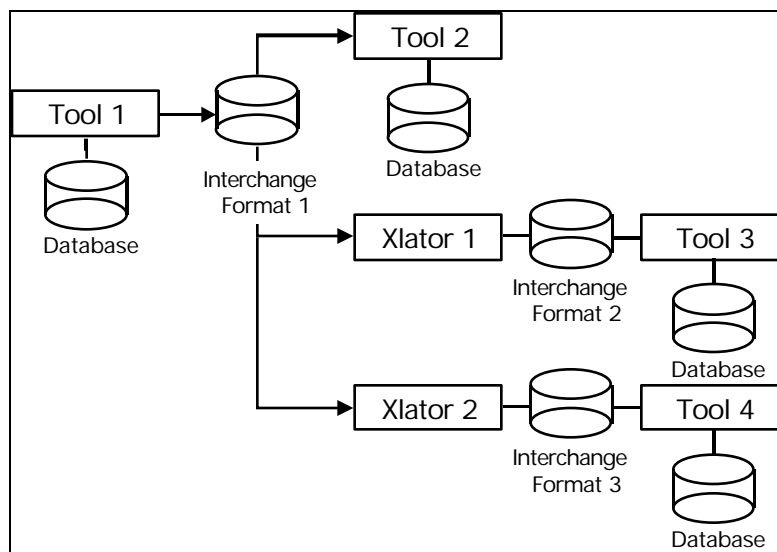


Figure 22: 6.1 File Centric EDA

As the number of tools grows and the size of the design grows, three problems arise:

- The number of tools that need to access any particular subset of the design data increases
- The size of the design databases explodes as a function of (design-complexity x number of representations x number of formats generated by translators)

Replacing the entire data by a tool as the result of only incremental changes becomes unduly costly

Using Component Object Interfaces (e.g. CORBA, COM, and Java Beans) ameliorates some of these problems because:

- There is one set of access code for each interface rather than for each tool x each format
- The size of the design database is proportional to (design-complexity x number of representations) and there only needs to be one format for each kind of data's external storage format. This saves

processing time because once a tool creates a datum, any tool can access it by calling the correct API without processing any unneeded data.

- Any incrementally updated data is immediately available to any downstream tools

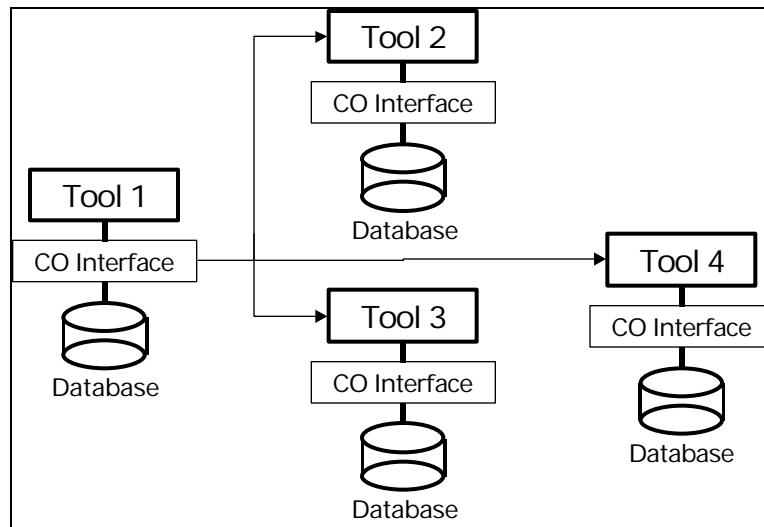


Figure 23: API Centric EDA

The second trend is based upon the tendency of CAD tool developers to use generic operating system services to the full extent possible, but not to allow the lack of such a service to limit the power of their tools. For example, early EDA layout editing tools had tool specific code for rendering the shapes in a layout, while most current tools use X-Windows or MFC code to perform these services. Access to design data has benefited from use of operating system services, but the access model will change as tools shift to the use of component objects.

Currently, EDA tools commonly communicate their data by exchanging files. Here, each tool records its resulting data into a file (in a de facto standard format, a public standard format or a proprietary format). This causes four problems:

- Each tool that needs to access the design data is required to parse the entire design file, and process all the data in the file (even if it only needs a small subset)
- If two tools that do not process the same file format need to exchange some design data, they need a translator program to map the data from one tool's format to the other's, after which the second tool will read it all again
- When a new version of a tool is released, it commonly has added new information or structure to its file format requiring either: (1) find a new revision of the receptor tools that accept the new format; (2), revise the translator to accept the new format; or, (3) write a new translator to map the new format
- Providing a multi-tool accessible file format imposes a heavy extra burden on its developers because: (1) they must provide documentation of the public format that is sufficient for people to write readers and writers in other tools; and, (2) they need to try to minimize changes in public file formats so as not to impose too heavy a maintenance burden on their users

Object Interfaces eliminate these problems:

- A tool that binds the interface can read only relevant data that it needs and not irrelevant data
- The provider of the interface maps the file data to the interface so each tool that binds the interface uses the same code to access the file data

- Interfaces are themselves versioned so if a tool changes a file format, it may support both the old version of the interface and the new version, which means that tools that depend upon the old format don't break when the new tool comes on stream

There are already a large number of support tools that help an interface supplier make it accessible to potential users and in addition, the versioning helps to reduce the impediment to evolving tools

The Taskforce feels that the expansion of EDA tools needs the kind of services that this component object architecture structure affords. The result will provide users suites of flexible tools, involving few translation steps, and evolve rapidly.

Recommendation VIII: EDA System Structure

- | |
|---|
| <ul style="list-style-type: none"> ▪ Semiconductor Process Changes <ul style="list-style-type: none"> - N/A ▪ Design Methodology <ul style="list-style-type: none"> - N/A ▪ Design Automation <ul style="list-style-type: none"> - Develop with Object Interfaces <ul style="list-style-type: none"> <i>Read only Relevant data</i> <i>Interface Maps from files to Objects</i> <i>Versioned Interfaces</i> <i>Multi-tool accessible formats</i> |
|---|

SECTION 7

A Vision of 100 nm Microprocessor Design

The Taskforce has identified future transitions required to design microprocessors. These transitions are not limited to microprocessors. Many other chip designs, like graphics controllers, share the same characteristics with other custom chips like graphic controllers. These chips will be built in silicon processes with geometries less than 100 nm and which will operate at GHz frequencies. Even though the Taskforce concentrated on microprocessor design, it is expected that each of the paradigm shifts will affect a wide range of designs.

It is apparent that there are multiple solutions to the paradigm shifts that the Taskforce has investigated. These solutions span processing, design methodology, and design automation. Some solutions will be preferable compared to others. Some challenges will be so difficult to control that multiple techniques will be required to alleviate their effects. Each individual paradigm shift is only an element of the whole. This section of the Roadmap examines the interrelationships between design issues, and how solutions relate and complement each other.

PROCESS ENHANCEMENTS

The Taskforce has identified signal integrity as a major challenge. Signal distortion will increase as a result of power transients and signal interference and many details arise concurrently to upset signals thus requiring they be re-timed. Further, to deliver hundreds of amps to sections of the circuit instantly, the signals on the bus lines need to be stable in a determined amount of time and not have bus signals with wide variations in delay.

It is not a simple matter of designing around signal integrity, as there are too many problem sources for that approach. An engineer cannot concentrate on details when there are billions of details therefore, the challenge is in the handling of details while efficiently doing design. A significant methodology change is needed to remedy the problem. The characteristic of the problem needs to change.

SIGNAL INTEGRITY

The most effective technique of controlling inter-signal interference is by shielding and the use of power planes to supply primary current to the chip. These techniques should be expected since they have been used in PWBs and MCMs for many years.

The upside benefit of adding layers of metalization is design efficiency. The downside is a significant increase in processing cost. However, other solutions to signal integrity require spreading the elements of a design further apart, which requires larger chip area and at substantial costs. Also, the di/dt on power lines, clock, bus and other long high fanout interconnects will be substantial, and the inductive as well as the mutual inductive effect will be dominating.

The Taskforce recommends that the one power distribution layer be expanded to as many as four layers of metal. Two of these layers will serve as power and ground planes and two layers will serve as interconnect shielding and block shielding. The power and ground planes should be adjacent to each other with a maximum capacitance between the layers. This will reduce power spikes but additional suppression will most likely be required.

The Taskforce recommends that process technology either build a large integrated capacitor into the chip, or use MCM technologies to attach an array of capacitors to the chip. The capacitance between power and ground will need to be capable of sustaining voltages for as long as a nanosecond while the power surges reach hundreds of amps. That implies that a large capacitance is required.

Recommendation IX: Process Modifications

- **Semiconductor Process Changes**
 - Power Delivery
 - Power and Ground Planes*
 - On chip and/or On MCM Bypass Capacitors*
 - Signal Integrity Assurance
 - Shielding*
 - Low mutual capacitance and mutual inductance materials*
- **Design Methodology**
 - N/A
- **Design Automation**
 - N/A

Soft Errors

Soft-errors are random nondestructive events resulting in recoverable circuit error, which are caused by the induced noise from subatomic particles or alpha particles resulting from radioactive decay of materials. When the prevalence of signal upset is high enough, processing must find a means to harden circuits against soft errors. Silicon-on-insulator (SOI) is one method for achieving hardening. SOI has, also, the potential of being a superior high frequency technology. The Taskforce recognizes the risks that conversion to SOI may entail, but suggests that it may be necessary and the technology must be ready.

METHODOLOGY ENHANCEMENTS

Chip design methodology will require even greater changes. The objective of design is the development of a chip or chip/system that is functionally working, and that meets performance requirements in a predictable amount of design time. 100 nm designs are more complex than previous generations because many details that will overwhelm design, because there will be more transistors to design and because designs will not converge without a strategic plan.

- The Taskforce recommends that designers use a meet-in-the-middle approach

This approach uses some rigid hierarchies as the basis for design. At the top of the hierarchy is architectural definition and at the bottom is physical layout and test. The middle is the full chip level where architectural design meets physical design.

- The Taskforce recommends more use of staggered clocks or asynchronous logic

The Taskforce believes that an increasing amount of design will be accomplished at the full chip level. This may include global optimizations such as use of staggered clocks or asynchronous logic or minimizing the number of signal transitions.

- The Taskforce recommends use of rule based design

A typical rule may be to include buffering and shielding in a cell to make it signal interference resistant. Verification at the full chip level is reaching the end of its useful life. At the full chip level, there are just too many things to verify. Prevention and techniques to avoid predicaments must become the approach. From a timing standpoint, timing delays and the delay variability will be determined at low levels of hierarchy and abstracted into forms that can be applied at higher levels. The same approach is necessary for power and transient power, for test and built-in means of testing at speed, and for physical characteristics such as area and pinout.

Global interconnect will dominate over transistor delays and design must become interconnect-centric. Because the longest interconnections are between blocks, this part of the physical design must be completed first. The rest of a design will flow from the decisions made at this level. To make these decisions, a design engineer will estimate the characteristics of functional blocks in an analogous manner to the way an engineer estimates interconnect today.

- The Taskforce recommends that low level design be considered a physical design task with constraints

In order to generate a block, its function is specified. Commonly these functions are synthesized or compiled into low level blocks that are then placed and routed. In GHz chips, however, physical design predominates over function. The function has degrees of freedom to allow the other constraints to be met. It is common to see similar system architectures be designed with different functional blocks. The methodology involves one-time verification at this level, then repeated use known-good pre-validated abstractions at other levels.

One of these constraints is a high level functional specification. At the completion of the low level design, a designer needs to build a higher level block-model that abstracts the detail but maintains the essential fidelity required to assemble blocks. It is at the block level where the full chip level will be assembled and verified. The Taskforce believes that design convergence is a function of attention paid to initial specification, attention to detail at the lowest level of hierarchy, and attention to abstracting the low level of detail (including physical back annotation) to a chip level block. As blocks require iteration, then the change should be tethered to the old circuit to reduce the magnitude of the verification effort.

Recommendation X: New Design Methodologies

▪ Semiconductor Process Changes
- N/A
▪ Design Methodology
- Signal Integrity Design Methodology
<i>Meet-at-the-full-chip level Design Approach</i>
<i>Hierarchical Design that is Interconnect-centric</i>
<i>Staggered Signals and Asynchronous Logic</i>
<i>Built-in Test</i>
- Rules Based Design
<i>Constraints</i>
<i>Top Down Forecasting</i>
<i>Bottom up Model Building</i>
▪ Design Automation
- N/A

High level decisions are a natural outcome of low level considerations. Low level development is driven by chip level budgets. These budgets will set the realizability of a design. So to begin a design process, an engineer will need to estimate or forecast the characteristics of the block being considered. The characteristics should include power requirements, built-in test, inter-block timing, and signal integrity factors. These factors will be assembled from archived designs and from modeling or prototyping efforts.

ELECTRONIC DESIGN AUTOMATION

New categories of design tools will be needed. These tools can be grouped into two categories. The first category is tools that improve a particular level of design. The second category is tools that improve inter-level design.

Captive design support activities and commercial EDA software companies have vigorously developed design automation in the first category. These EDA activities are numerous and they often invent new solutions for particular design problems. A few examples of needed tools in this first category are:

- Power and supply current analysis must take into account: average as well as instantaneous surges. This analysis must determine how high power will affect signal integrity and thermal variation in delay.
- Multi-path with multi-signals timing analysis. Because delay is a function of layout as well as relationships between signal directions, a new class of timing analyzers is required to verify circuit operation.
- Placement/Route is one way to control signal integrity and power, and to minimize the number of unwanted signal interactions. Placement/Route automation needs to support noise immune technologies such as exploitation of shielding, repeater/receiver hysteresis and transition slew rate control.
- Physical design generators for arithmetic as well as other functions need to combine synthesis/placement/routing for inter-block connections that are signal-integrity sensitive that exploits other techniques to improve noise margins. They provide rapid regeneration of pre-designed blocks (cores) using constraints for aspect ratio, timing, power, noise, etc.
- Microprocessor design will increasingly become a large software project coupled with a very regular and quite disciplined hardware design project. New code generators are needed to enable the hardware to make better use of regular structures. The design approach needs to be switched from local repair of problems to global avoidance.

The Taskforce feels that the industry will naturally find solutions for these real challenges. EDA has been capable of generating this class of solutions in the past this trend will undoubtedly continue. However, it is the second category, improving inter-level design, where EDA has not been generally successful. Inter-level implies multi-discipline, which often necessitates the integration of software that was never intended to co-exist.

As part of preparatory tasks, the Taskforce reviewed results from previous taskforces on EDA. Each one identified tool integration as the most important task ahead and yet, EDA tool integration has been met with very limited success.

- This Taskforce recommends that the picture of integration be changed. As long as each EDA sphere orbits at its own rate, alignment will be infrequent. Because there are billions of elements of data, EDA software structure must be built on a strong base of information flow with strict alignment between the tools in the flow. Thus, alignment must somehow be designed into the system. Alignment between design tools and thus EDA companies, for chip design requires a concerted methodology. The Taskforce is pessimistic that EDA tools can be interfaced into alignment and we are pessimistic that EDA be assembled into alignment. EDA has created spheres of excellence, which the Taskforce feels can be molded into alignment only through the use of Object Interface design.
- The Taskforce recommends that all EDA become interconnect-centric. The Taskforce further recommends that EDA software architectures incorporate a new set of tools. These tools aim at the same interconnect-centric goal from another view. The Interconnections that are most dominant are the longer lines that interconnect blocks or carry global signal such as clocks and busses. These interconnections are so important that they need to be known, minimized, and analyzed before defining lower levels of detail. They determine delay and skew between events and are the

largest single power consumers. Further, their length and proximity to other interconnect and active devices causes the greatest level of signal integrity problems.

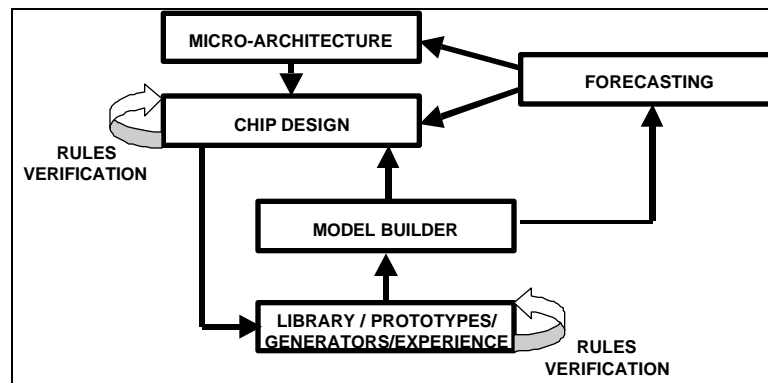


Figure 24: Interconnect-Centric EDA

- For interconnect to be designed, there has to be a set of functional blocks to connect. Therefore, the design process will initiate with estimation of the blocks that will later undergo detail design. The Taskforce suggests that EDA create an ability to forecast block design. This capability will be built upon libraries, archived design data, synthesizers and generators. This is a special case of rules driven design where each class of blocks will adhere to rules. For example:

IF *blocktype* = MEMORY, THEN DO ____;
 IF *blocktype* = REGISTER STACK, THEN DO ____;
 IF *blocktype* = DECODER, THEN DO ____.

The forecaster value will depend on the design activity's ability to collect knowledge and build models. The models will map lower level detail into high level design. The high level design must deal with models for signal interference, block timing with uncertainties, and power consumption. The model builders will need to extrapolate from libraries, prototyping, or generators.

- The Taskforce recommends that verification be built into the design process. Rule base audits are very effective. An auditor queries implementations as they are created to avoid problems. It will interrogate the full range design space: signal integrity, power, testing, timing, and area compliance. Difficult situations will be eliminated before they cause major design iterations.

In summary, this Taskforce dares to ask software to begin again. For those who question if this is needed, let us examine EDA history. What happened to: Tegas, Calma, Daisy, Valid, Crosscheck, Scientific Calculations, and Silvar-Lisco. EDA is change. The only question is who will fill the void.

Recommendation XI: New EDA

- **Semiconductor Process Changes**

- N/A

- **Design Methodology**

- N/A

- **Design Automation**

- New Design Tools

- Power Analyzer Considering Signal Integrity and Thermal variation in Delay*

- Multi-path/signal Timing Analysis*

- Integrated Synthesis, Placement and Routing that Controls Signal Interactions and Power Problems*

- BIST*

- New Design System

- Object Interfaces*

- Forecaster and Estimator*

- Rules Verifier*

- Model Builders*

SECTION 8

Future Research

The Taskforce has encountered a number of challenges that require further research.

SOFT-ERRORS

Soft-errors are random non-destructive events resulting in recoverable circuit error, which are caused by the induced noise from subatomic particles. The perturbation caused by the particles interacting with silicon atoms can cause the value of a storage cell to change if the induced charge goes above or below a fixed level ($Q_{critical}$). Further, transient output pulses in logic circuits may indirectly produce changes in the state of other circuits if they occur at critical time periods, such as during clock or data transitions. There are two sources of these subatomic particles: alpha particles and cosmic particles.

Alpha particles are high-mass particles that result from the radioactive decay of materials within the IC or its contacts. Alpha particles can produce a great amount of ionization along their track, thus liberating large numbers of electron-hole pairs from atoms over a very short distance. If free electrons are captured on a capacitor charge plate, a single event upset (SEU) may occur resulting, for example, in the memory-state may be flipped. Additionally, transient output pulses may result on signal lines that indirectly produce changes in the state of circuits if they occur at critical time periods, such as during clock or data transitions.

Cosmic particles are high-energy particles that emanate from outer space. High energy cosmic particles can fracture the nucleus of silicon atoms thus causing a stream of secondary particles that can produce a great amount of ionization, thus liberating large numbers of electron-hole pairs from atoms over a very short distance. If free electrons are captured on capacitor charge plate, the memory-state may be flipped. Again, transient output pulses may result on signal lines that indirectly produce changes in the state of circuits if they occur at critical time periods, such as during clock or data transitions.

The probability of a soft-error occurring on an IC is a function of several factors making it difficult to quantify:

- The materials' properties, the altitude at which the IC must operate, and shielding each affect the probability of a particle strike.
- The feature size and feature packing, as well as the design itself affect the probability that a strike will cause an event upset (or soft-error).

Whether or not an occurring soft-error impacts the execution of the IC is a function of whether or not the bad state is used before it is refreshed to a correct state.

To understand the impact of feature size on the probability of soft-errors, the following should be considered. The charge of a memory device is approximated by $Q=CV$, where C is the gate capacitance and V is the voltage differential. Gate capacitance is approximated by $C=(\epsilon_{GOX}A_{gate})/T_{GOX}$, where ϵ_{GOX} is the dielectric constant for the gate, A_{gate} is the gate area (which scales as the square of the feature size), and T_{OX} is the oxide thickness (which scales with feature size). Thus, the gate capacitance (given no change in materials properties) decreases proportionally with feature size.

The relationship of charge to the Technology Roadmap is given in Figure 25. As shown, the state charge for latches and RAM decreases by 62.6% by the year 2003, as a result of decreasing feature size and V_{dd} . Since the charge on an electron is constant, this implies that the relative number of free electrons resulting from ionization decreases by the same percentage. Therefore, as an approximation (given no changes as a result of materials properties), the probability that a particle strike will cause an event upset might be expected to increase by the same proportion.

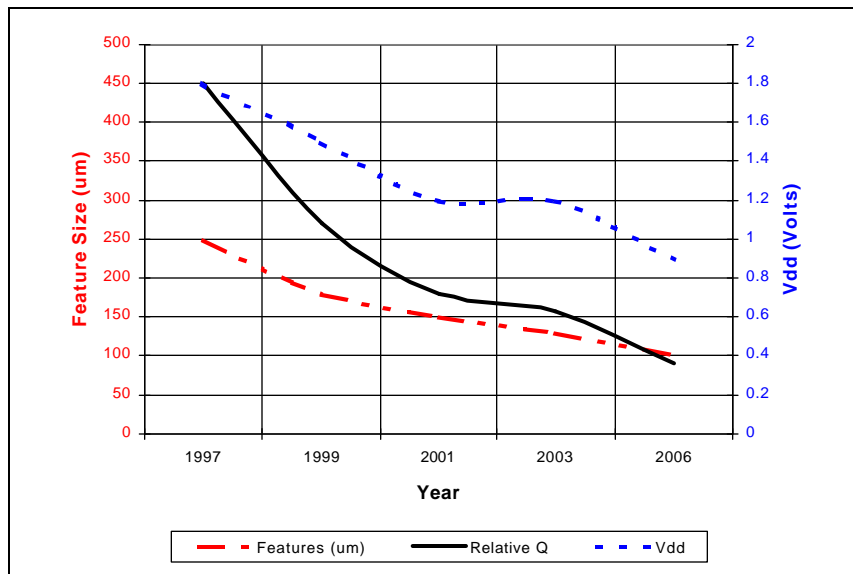


Figure 25: CMOS Charge is Decreasing

Prevention of particle strikes may be improved by the use of materials within the IC that result in less radioactive decay and by shielding the IC. Shielding the IC from cosmic particles is not practical given the extremely high energy levels. Shielding transistor gates from the free electrons that result from particle strikes may be improved by process technology such as silicon-on-insulator (SOI). With SOI, the oxide insulator can reduce the number of free electrons that are in close enough proximity to the gate to be swept up.

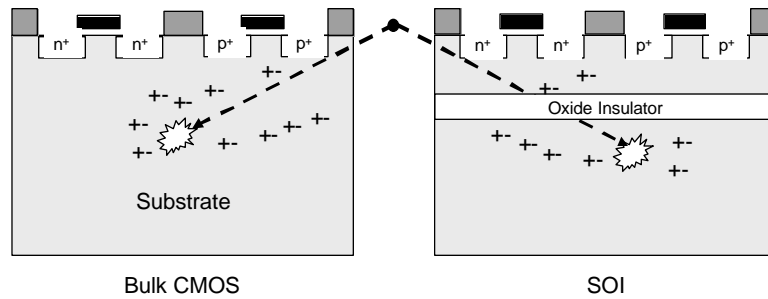


Figure 26: Soft Errors in CMOS and SOI

Detection and Correction must necessarily be designed in since soft-errors, by definition, are not hard physical defects. Therefore, it is expected that there will be increased use of parity checking and modular redundancy in microprocessor designs. Additionally, there will be more use of error correcting schemes that can correct for both single and multiple-event upsets within a functional unit—such as a memory word or register.

The probability of a soft-error that affects machine operation is extremely small, but it is increasing as ICs shrink. Even with a low probability, it is an area that deserves attention as a single error occurrence could cause the machine (using the IC) to crash or produce incorrect results. The Taskforce has evaluated several studies and has concluded that the exposure to soft errors could not be quantified, therefore further research is recommended. The following list of EDA research and development may be necessary based on the results of such research:

Detection/Correction Logic Insertion

Similar to the productivity advantages to supplying scan-register insertion tools in today's EDA systems, in the near future, automatic insertion of soft-error detection logic will be desirable. Quite possibly, this

support would allow the designer to identify susceptible functions (or regions) within a design and the detection technique (parity, modular redundancy, etc.) to be used, and allow computer automation to insert the correct detection circuitry.

Similarly, research into computer automation to insert correctly functional logic to process error correction techniques (with standard techniques such as Hamming codes, etc. available to the user) is desirable.

Physical Design

100 nm physical design tools will maintain a complex model of the rule-constrained and budget-limited physical design hierarchy elements. For each property managed in service of known-correct overall chip design, the physical tools may (for efficiency's sake) be required to compute the incremental differences in properties that are consequences of proposed local design changes. The specific requirements for such tools remain to be determined.

It is also likely that physical design considerations may affect the probability of multiple soft-errors within a design. Physical separation of circuits from their redundant copy or bits in a register may be an important consideration for effective placement algorithms. The density of latches and memory in regions of the IC will be important to understand such diverse factors as peak pulse power needs and the overall susceptibility to multiple-event upsets.

Modeling Tools

Research has shown that the critical charge level ($Q_{critical}$) is a function of the variances in a number of variables such as doping levels, spacing, and current levels. Further research and development of accurate and efficient modeling and simulation tools that can predict the probability distribution of soft error susceptibility across regions of the chip will be required. These tools must consider the manufacturing process variations, feature size and spacings, and frequency, etc.

ASYNCHRONOUS DESIGN

With the rise in power and especially with the instantaneous current surges that occur at clock edges, it will be increasingly important to disperse circuit switching times. One approach to reduce these periodic surges is asynchronous design. Asynchronous design has fallen into disuse because synchronous design has been greatly automated, while asynchronous design remains both more challenging and less well supported by tools. Research is needed to automate asynchronous design, making this design style both low-risk and relatively painless. A different approach to timing analyzers will be required because most presently deployed timing tools presume clocked synchronous design practices. Asynchronous synthesis will require enhancement and asynchronous testing approaches may need revision.

MEASURE DESIGN EFFECTIVENESS

As the Taskforce evaluated design data, it became apparent that design efficiency and productivity are not measured, and that management of productivity has been in most cases ad hoc. Those who report productivity experience a broad range of results. Often, one organization's productive environment will be less effective than another organization's unproductive design. It appears that the underlying issues go far beyond differences in expectations and perceptions. Methods are needed to measure 100 nm design team productivity. Likewise, measures of quality that determine how a design compares to criteria that are created for a process and machine architecture are needed.

Measures such as the number of transistors designed per engineering day have been used. However, when such factors are low, the value of the measure is questioned. Design complexity or other external factors often excuse poor performance. If a design team is rated by a factor, many managers will optimize the outcome for that factor. It should be an objective to use these factors to enhance design effectiveness and not manage an ongoing process. For that reason, a controlled experiment at the university level followed by industrial appraisal is preferable.

Multiple universities can be funded to design the same microprocessor. The microprocessor will be specified as a behavior and the design team's task will be to complete a chip design when investigating

methodologies and automation techniques. In parallel with the design teams, a team of industrial psychologists who will observe the process and factors such as task times, individual times, group times, strategies, successes, failures, iterations, along with EDA ability to perform effectively during each phase of a design. The design quality can be measured through chip size and speed along with time to design (equivalent to time to market). The factor will be drive future enhancements that will advance design effectiveness.

Recommendation XII: Further Research

- **Semiconductor Process Changes**
 - Soft-error Prevention
- **Design Methodology**
 - Measures of Design Effectiveness
- **Design Automation**
 - Physical Design for Soft-error Reduction
 - Asynchronous Design Automation

SECTION 9

FUTURE TASKFORCE ACTIVITIES

The EDA Roadmap Taskforce has investigated high-end microprocessor design. It considered other market segments as well, but concluded that attempting to analyze multiple market segments together may lead to different conclusions than focus on only one would yield. Two markets that the Taskforce feels are important to study, however, are High-End Core Based ASICs and Battery Powered Communications Products. It is recommend that two additional committees be assembled simultaneously to study both.

When staffing these committees the most important factor is identifying individuals to lead the committees with industrial backgrounds that are heavily weighted toward theoretical work. The leaders should have previously demonstrated an ability to lead volunteer organizations. These individuals must be personally committed to the technology, the Taskforce objectives, and to the other volunteers. They should build very active committees that meet physically together monthly. This Taskforce found the greatest attendance at meetings that were held in the Silicon Valley. Even the attendees that resided outside this area were more prone to attend meetings held in Silicon Valley. The sponsors must be willing to cover expenses of the committee's operation.

Recommendation XIII: Future Taskforce Activity

- | |
|--|
| <ul style="list-style-type: none">▪ Future Taskforce Activities<ul style="list-style-type: none">- Battery Powered Communications- High-End Core Based ASICs |
|--|

Appendix A: Acronyms

Source: NTRS97

A

AC, alternating current

API, application program interface

ASIC, application-specific integrated circuit

ATE, automatic test equipment

B

BiCMOS, bipolar complementary metal-oxide semiconductor

BIST, built-in self test

BIT, built-in test

BW, bandwidth

C

CAD, computer-aided design

CMOS, complementary metal-oxide semiconductor

COM, Common Object Model

CORBA, Common Object Request Broker Architecture

CPU, central processing unit

CTE, coefficient of thermal expansion

D

D&T, Design and Test

DARPA, Defense Advanced Research Programs Agency

DC, direct current

DFM, design for manufacturability

DFT, design for test

DLT, device level test

DoD, Department of Defense

DRAM, dynamic random access memory

DSM, deep sub-micron

DSP, digital signal processing

DUT, device under test

E

ECAD, engineering computer-aided design

ECC, error-correcting circuitry

ECL, emitter coupled logic

EDA, electronic design automation

EDAC, Electronic Design Automation Consortium

EDI, electronic data interchange

EDIF, electronic design interchange format

EM, electromigration

EMC, electromagnetic compatibility

EMI, electromagnetic interference

ESD, electrostatic discharge

F

FPGA, field programmable gate array

FSM, finite state machine

G, H

GHz, GigaHertz

GUI, graphical user interface

HDL, hardware description language

I

IC, integrated circuit

IDDQ, direct drain quiescent current

I/O, input/output

IP, ion projection OR intellectual property

IR, infrared

J, K, L

LSI, large-scale integration

M

MC, Monte Carlo model

MCM, multichip module

MLM, multi level metal

MOS, metal-oxide semiconductor

MOSCAP, metal-oxide semiconductor capacitor

MOSFET, metal-oxide semiconductor field effect transistor

MPU, microprocessor

N

Nm, nanometers

NMOS, negative channel metal-oxide semiconductor

Noise, introduction of unwanted voltage on a signal or power line

NSF, National Science Foundation

NTRS97, National Technology Roadmap for Semiconductors, 1997 Edition

O

OPC, optical proximity correction

OTA, overall timing accuracy

P

PMOS, positive channel metal-oxide semiconductor

PPC, process proximity correction

PSM, phase shift mask

PWB, printed wiring board

Q, R

R&D, research and development

RAM, random access memory

RF, radio frequency

RMS, root mean square

RTL, Resistor Transistor Logic OR resistor transistor level

R-L-C-M, resistance-inductance-capacitance-mutual (inductance)

S

S/D, source/drain

SEU, Single Event Upset

SEMATECH, Semiconductor Manufacturing TECHNOLOGY

SEMI, Semiconductor Equipment and Materials International

SI2, Semiconductor Integration Initiative

SIA, Semiconductor Industry Association

SMT, surface mount technology

SOC, system on a chip

SOI, silicon on insulator

SRAM, static random access memory

SRC, Semiconductor Research Corporation

T

TCAD, technology computer-aided design

TPG, test pattern generation

U

ULSI, ultra large scale integration

UV, ultraviolet

V

VHDL, VHSIC hardware descriptive language

VLSI, very large-scale integration

VSI, Virtual Sockets Interface

V_{dd} , Power Source Voltage

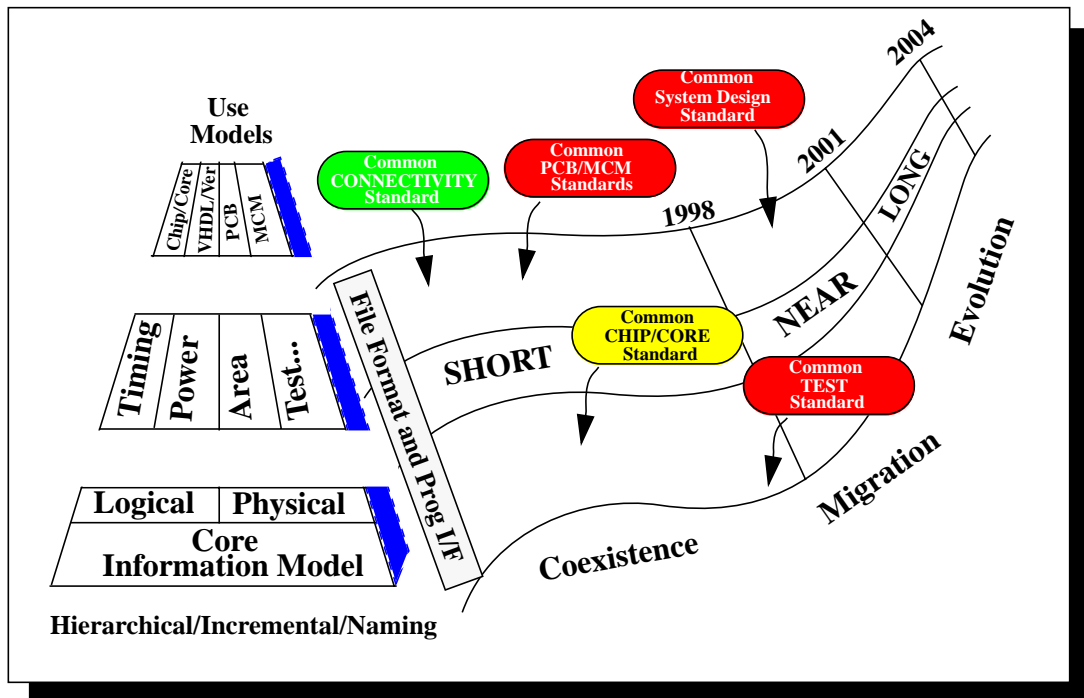
V_T , Threshold Voltage

W, X, Y, Z

Appendix B: References

- [1] Semiconductor Industry Associates, *National Technology Roadmap for Semiconductors*, 1997, www.sematech.org
- [2] EDA Industry Council, www.si2.org/ic
- [3] EDAC, Electronic Design Automation Consortium, 111 West Saint John Street, Suite 200, San Jose, California 95113, Phone: (408) 287-3322, www.edac.org
- [4] SEMATECH, 2706 Montopolis, Austin, TX, Phone (512) 356-3500, www.sematech.org
- [5] SRC, Semiconductor Research Corp, P.O. Box 12053, Research Triangle Park, NC 27709-2053, Phone: (919) 941-9400, www.src.org
- [6] Silicon Integration Initiative, Inc., 4030 West Braker lane suite 550, Austin, TX 78759, Phone (512) 342-2244, www.si2.org
- [7] DARPA, Defense Advanced Research Projects Agency, www.darpa.mil

EDA Industry Standards Roadmap - 1996



EDA Industry Standards Roadmap

EDA Industry Standards Working Groups:
EDA Integration and Interoperability Working Group
Design and Data Management Working Group
Technology Libraries and Models Working Group

Version 1.0 - 011596

Copyright © 1995, 1996 by CFI, Inc. All rights reserved worldwide. Permission to use and reproduce this documentation for internal purposes is granted under the following conditions. No right is granted to modify or resell this documentation for commercial purposes without specific, prior written permission from the CFI, Inc. All reproductions shall include this copyright notice and permission notice. CFI, Inc. provides this publication “as is” without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability or fitness for a particular purpose. CFI, Inc. may revise this publication from time to time without notice.

CFI, Inc.
4030 West Braker Lane, Suite 550
Austin, Texas 78759
USA

Phone: 512-338-3739; Fax: 512-338-3853; email: cfi@cfi.org

Windows is a registered trademark of Microsoft Corporation.
OS/2 is a registered trademark of IBM Corporation.
UNIX is a registered trademark of UNIX System Laboratories.
X/Open is a registered trademark of X/Open Company, Limited.
OSF is a trademark of the Open Software Foundation.

The development of this document was initiated by CFI, EDAC, and SEMATECH to address the standards challenges of the EDA industry for the next decade. This document was produced using FrameMaker® workstation publishing software provided by Frame Corporation.

Table of contents

Document Access and Updates	2
On-line Access	2
About This Book	3
Audience for this Book	3
How this Book is Organized	3
How to Find the Information You Want	4
Glossary of Standards	4
1 Introduction	5
1.1 Charter	5
1.1.1 Identify the EDA Industry Requirements on EDA Systems	5
1.1.2 Review Status and Current Plans of Related Standards	5
1.1.3 Identify Standards Areas Requiring Improvement	6
1.1.4 Determine How to Coexist and Migrate to Improved Standards	6
1.1.5 Develop a Roadmap to Future Standards	6
1.1.6 Deliver Recommendations and Roadmap Contributions	6
1.1.7 EDA Standards Industry Council	7
1.2 Background	8
1.2.1 Productivity Improvement	8
1.2.2 Complexity Management	8
1.2.3 Advanced Technology Development	9
1.2.4 Technology Development Funding	9
1.3 Scope	9
1.3.1 Electronic Design and Test Standards Focus	10
1.3.2 Technology Packages	11
1.3.3 Design Phases	11
1.3.4 Key Electronic Design and Test Interfaces	11
2 Executive Summary	13
2.1 Roadmaps	13
2.1.1 Introduction to the Roadmap	13
2.1.2 Overview of Roadmap Categories	14
2.1.3 Design System Roadmaps	15
2.1.4 Design Information Roadmaps	16

2.2 Recommendations	17
2.2.1 Key Roadmap Recommendations	17
2.2.2 Coexistence and Migration Strategy	21
2.2.3 Areas of Convergence	21
2.2.4 Areas of Acceleration of Work	23
2.2.5 Areas Where New Standards Work is Required	24
2.2.6 Areas Where Additional Roadmap Work is Required	24
2.3 The Standards Development Process	25
2.3.1 Current Standards Development Environment	25
2.3.2 Standards Development Process Recommendations	25
3 Electronic Design and Test Environment	31
3.1 Emerging Paradigm Shifts	31
3.1.1 Innovation in Systems Level Design (Architecture and High Level Design)	31
3.1.2 Innovation in Design Process Management	31
3.1.3 Increased Codesign Across Design Disciplines	31
3.1.4 New Architectural and Integration Concepts	32
3.1.5 Changing Business Practices	32
3.1.6 Pay-Per-View for Design Tools	32
3.1.7 Object Oriented Software Development	33
3.2 Pressures on Designers and CAD Integrators	33
3.2.1 Exploit Multiple EDA Operating Environments	33
3.2.2 Use Diverse Databases and Formats	33
3.2.3 Use Tools from Multiple Tool Vendors	33
3.2.4 Enforce Design Methodologies and Process Management	34
3.2.5 Reduce (or Maintain) Cycle Time	34
3.2.6 Reduce Design Costs	34
3.2.7 Maximize Return-on-Investment (Price/Performance)	34
3.2.8 Design for Quality	35
4 The Design System (Infrastructure and Tools)	37
4.1 Computing Environment and User Interface	37
4.1.1 Current Environment - Computing Environment and User Interface	37
4.1.2 Requirements - Computing Environment and User Interface	38
4.1.3 Recommendations - Computing Environment and User Interface	39
4.1.4 Roadmap - Computing Environment and User Interface	40
4.2 Design Tool Communication	41
4.2.1 Current Environment - Design Tool Communication	41
4.2.2 Requirements - Design Tool Communication	41
4.2.3 Recommendations - Design Tool Communication	41

4.2.4 Roadmap - Design Tool Communication	42
4.3 EDA System Extension Language	42
4.3.1 Current Environment - EDA System Extension Language	42
4.3.2 Requirements - EDA System Extension Language	43
4.3.3 Recommendations - EDA System Extension Language	44
4.3.4 Roadmap - EDA System Extension Language	45
4.4 EDA Standards Based Software Development Environment	45
4.4.1 Current Environment - Standards Based Development Environment	45
4.4.2 Requirements - Standards Based Development Environment	46
4.4.3 Recommendations - Standards Based Development Environment	46
4.4.4 Roadmap - Standards Based Development Environment	46
4.5 Intellectual Property Protection for Design Data Objects	50
4.5.1 Current Environment - Design Data Object Protection	50
4.5.2 Requirements - Design Data Object Protection	50
4.5.3 Recommendations - Design Data Object Protection	50
4.5.4 Roadmap - Design Data Object Protection	50
4.6 Design Management	51
4.6.1 Current Environment - Design Management	51
4.6.2 Requirements - Design Management	52
4.6.3 Recommendations - Design Management	53
4.6.4 Roadmap - Design Management	53
4.7 Design Data Management	54
4.7.1 Current Environment - Design Data Management	54
4.7.2 Requirements - Design Data Management	56
4.7.3 Recommendations - Design Data Management	56
4.7.4 Roadmap - Design Data Management	57
4.8 Design Process Metrics Management	57
4.8.1 Current Environment - Design Process Metrics Management	57
4.8.2 Requirements - Design Process Metrics Management	58
4.8.3 Recommendations - Design Process Metrics Management	58
4.8.4 Roadmap - Design Process Metrics Management	58
4.9 Design Tool Management	59
4.9.1 Current Environment - Design Tool Management	59
4.9.2 Requirements - Design Tool Management	60
4.9.3 Recommendations - Design Tool Management	61
4.9.4 Roadmap - Design Tool Management	62

4.10 Resource Management	63
4.10.1 Current Environment - Resource Management	63
4.10.2 Requirements - Resource Management	63
4.10.3 Recommendations - Resource Management	65
4.10.4 Roadmap - Resource Management	65
5 The Design Information (Design Data Representation)	67
5.1 Common Topics Across Design Information	67
5.1.1 Incremental Processing	67
5.1.2 Hierarchical Processing	69
5.1.3 Design Object Naming	70
5.2 Common Topics Across Design Steps	71
5.2.1 Timing Information	71
5.2.2 Simulation and Test Control	76
5.3 System Level Design	78
5.3.1 Current Environment - System Level Design	78
5.3.2 Requirements - System Level Design	78
5.3.3 Recommendations - System Level Design	81
5.3.4 Roadmap - System Level Design	82
5.4 Detailed Design	83
5.4.1 Current Environment - Detailed Design	83
5.4.2 Requirements - Detailed Design	83
5.4.3 Recommendations - Detailed Design	89
5.4.4 Roadmap - Detailed Design	90
5.5 Design and Technology Re-use	92
5.5.1 Environment - Design and Technology Re-Use	93
5.5.2 Requirements - Design and Technology Re-Use	94
5.5.3 Recommendations - Design and Technology Re-Use	97
5.5.4 Roadmap - Design and Technology Re-Use	97
6 Key Interfaces to Other Domains	101
6.1 Manufacturing Build Interface	101
6.1.1 Current Environment - Manufacturing Build Interface	101
6.1.2 Requirements - Manufacturing Build Interface	102
6.1.3 Recommendations - Manufacturing Build Interface	102
6.1.4 Roadmap - Manufacturing Build Interface	103
6.2 Manufacturing Test Interface	103
6.2.1 Current Environment - Manufacturing Test Interface	103

6.2.2 Requirements - Manufacturing Test Interface	104
6.2.3 Recommendations - Manufacturing Test Interface	106
6.2.4 Roadmap - Manufacturing Test Interface	107
6.3 Mechanical Design Interface	108
6.3.1 Current Environment - Mechanical Design Interface	108
6.3.2 Requirements - Mechanical Design Interface	109
6.3.3 Recommendations - Mechanical Design Interface	109
6.3.4 Roadmap - Mechanical Design Interface	109
6.4 Software Design Interface (Hardware/Software Co-design)	110
6.4.1 Current Environment - Hardware/Software Co-Design	110
6.4.2 Requirements - Hardware/Software Co-Design	111
6.4.3 Recommendations - Hardware/Software Co-Design	112
6.4.4 Roadmap - Hardware/Software Co-Design	113
Appendix A - The Roadmap Development Team	A-1

List of Tables

Table 2.1: Areas of Recommended Standards Convergence.....	23
Table 2.2: Areas of Recommended Standards Acceleration	23
Table 2.3: Areas of Recommended New Standards Work	24
Table 2.4: Areas of Recommended Additional Roadmap Development.....	24
Table 5.1: Impact of Design Size on Design Processing Times	85
Table A.1: EDA System Interoperability and Integration Working Group (EII)	A-2
Table A.2: Design and Data Management Working Group (DDM)	A-4
Table A.3: Technology Libraries and Models Working Group (TLM).....	A-5

List of Figures

Figure 2.1— Design System Roadmap.....	15
Figure 2.2— Design Information Roadmap	16
Figure 2.3— The EDA Industry Standards Roadmap	17
Figure 2.4— Vision of Standards Development.....	27
Figure 4.1— Open EDA Enterprise Architecture.....	48
Figure 4.2— Open EDA Data Interoperability Architecture.....	49

Foreword

The *National Technology Roadmap For Semiconductors* (NTRS) makes a compelling argument for concern regarding the ability of Electronic Design Automation (EDA) tools to meet the technological advances that are predicted for integrated chips. Similar concerns are raised within the *National Technology Roadmap for Electronic Connections*. Both of these roadmaps assert the need for focused industry cooperation towards a common vision of development and design through pre-competitive research and development as well as standardization in several areas relating to design tools. To meet the challenge of EDA standards, CFI, EDAC, and SEMATECH initiated an industry-wide activity to develop this *EDA Industry Standards Roadmap*.

This Roadmap is the product of cooperation across all sectors of the EDA industry, including semiconductor companies, electronic equipment companies, commercial EDA companies, government organizations, standards organizations, and industry consortia. It provides a common structure for resource investment in the development of standards related to EDA in order to meet the increasingly complex technology needs of the electronics industry. Particular attention has been placed on the requirements of integrated circuit technology expansion; however, the demands of packaging technology are considered as well.

The Roadmap represents the consensus of the nearly fifty direct contributors as well as a multitude of industry professionals who provided crucial input on early drafts of the document. This Roadmap is not the end, but rather the beginning. The vision within the Roadmap must be seriously considered by the industry and, in order to avoid a sense of crisis management, its recommendations must be acted upon in a planned, evolutionary fashion. The Roadmap must also be enhanced and prioritized in a number of design areas that have not yet been given due attention. This includes packaging technology above the chip level as well as the interfaces between electronic design and other domains such as manufacturing (build and test), mechanical design, software design, and TCAD. Consideration for business paradigm shifts and their impact on EDA software distribution and quality requirements should also be considered more fully. Finally, and perhaps most importantly, the Roadmap should be re-evaluated at a regular frequency to assure that it matches current technological demand.

A great deal of gratitude is owed to the individuals that represented the Working Groups that developed the Roadmap and to ARPA for their financial support and encouragement. Very special recognition is deserved by John Teets, who assumed the role of Chief Technical Leader and Roadmap Editor/Writer. His tenacity and dedication, as well as his numerous technical contributions, made this Roadmap a reality.

Implementation of this Roadmap is now the responsibility of the EDA Industry Council (IC) and the companies and organizations it represents. CFI, EDAC and SEMATECH are committed to support this implementation. It is offered for consideration in hopes that it will serve as a significant planning guide for development and purchase of EDA tools, as well as standards support. If properly used and cultivated, the Roadmap will help industry to maintain the historical trends of technological and market growth.

Document Access and Updates

The EDA Industry Standards Roadmap document was prepared by the EDA Interoperability and Integration Working Group (EII), Technology Libraries and Models Working Group (TLM), and Design and Data Management Working Group (DDM) for review and approval by the EDA Standards Industry Council (IC). The Industry Council approved the 10/26/95 version of the Roadmap for immediate release as Version 0.9. This document reflects updates based on the feedback on Version 0.9 received by the end of December, 1995. The information presented in this document is for the purpose of discussion by the EDA industry at large, and feedback is welcome.

All feedback should be sent to CFI at “roadmap-feedback@cfi.org”.

On-line Access

An on-line version of this document is available at the CFI server via common web browsers such as Mosaic or Netscape, and a postscript version of the document is available via FTP.

World-Wide Web

To view on the web, Open URL to <http://www.cfi.org/roadmap/roadmapHomePage>, or Open URL to <http://www.cfi.org/ic> to access the Industry Council home page.

You can send direct feedback or comments on this online document by pressing the feedback button or you can send email directly to “roadmap-feedback@cfi.org”.

FTP Access

To download the complete postscript file of the actual document, ftp to cfi.org at <ftp://cfi.org/public/Cfi/Development/Roadmap/EII/roadmap.ps>

About This Book

The *EDA Industry Standards Roadmap* is the result of an enormous effort by the EDA community to refine by consensus the key EDA industry standards needs and recommended solutions for the key challenges of the next decade. The list of contributors is extensive. The numerous workshops, meetings, edit sessions and reviews, mostly via email, were time-consuming, yet were critical to the development of this Roadmap.

Audience for this Book

The information in this book is intended for several audiences:

- The EDA Standards Industry Council, who have reviewed and approved the contents, and who will provide the primary driving force for the EDA industry to implement the recommendations contained in the book
- EDA Integrators and Users, who are the primary *customers* with the key requirements to develop electronic products
- EDA Vendors and internal proprietary tool developers, who represent the primary EDA industry tool providers
- EDA standards groups, who need industry consensus input on standardization requirements and priorities
- University and Research, who provide additional leading edge research and development in support of the EDA industry.

How this Book is Organized

This book contains six chapters, summarized below:

- Chapter 1, "Introduction" describes the charter of this Roadmap, the three working groups that contributed to the Roadmap, and the scope of the work.
- Chapter 2, "Executive Summary" summarizes the key messages to the IC and the EDA industry regarding the requirements and the status of EDA standards with respect to those requirements, the recommendations for standards convergence, acceleration, and new standards work, and the recommended standards roadmaps for each category of requirements. The Executive Summary is designed to summarize the information in the chapters that follow it by expanding upon the electronic design and test area, and by discussing the environment, the requirements, the recommendations, and specific roadmaps in more detail.
- Chapter 3, "Electronic Design and Test Environment" reviews environmental topics that are relevant to the effective design of complex electronic systems. Also, key emerging paradigm shifts in the EDA industry and many of the pressures on design and CAD integrator and EDA tool developer teams are identified and discussed.

- Chapter 4, "The Design System (Infrastructure and Tools)", includes the computing environment and user interface, design tool communication, extension language, software development environment, and the design and data management areas. These topics are primarily domain-independent topics, but with an EDA focus.
- Chapter 5, "The Design Information (Design Data Representation)" addresses the key standards related to design data representations for all key design activities, including system level design (architectural and high level design activities) and detailed design (both logical and physical design in a given technology), as well as preparation for manufacturing build and test.
- Chapter 6, "Key Interfaces to Other Domains" discusses the interfaces to other design (or co-design) disciplines, and the key standards that relate to those interfaces. Manufacturing build and test interfaces, software interfaces, and mechanical design interfaces are also discussed.
- Appendix A, "The Roadmap Development Team" identifies the many people from across the worldwide EDA industry who participated in the Working Groups that developed this Roadmap.

How to Find the Information You Want

Below are some quick access tips to help you find the information you want to read about in this book.

- To access the *key roadmap recommendations*, their priority and timeframe, read Chapter 2, "Executive Summary".
- To access a given topic's detailed information including the *environment, requirements, recommendations and roadmap tasks* with descriptions, find the topic in the Table of Contents and go to the corresponding page.
- Those topics that relate to *Design and Data Management* are located in Chapter Four, since they are related to the general design system environment.
- Topics related to *Technology Libraries and Models* are located in Chapter Five in the section entitled "Design and Technology Reuse." These topics relate to specific reusable representations of EDA design data.

Glossary of Standards

A glossary of EDA industry standards has been developed and is available on-line via the CFI home page at <http://www.cfi.org/>. This glossary has been placed online to allow for frequent updating to add additional EDA standards, as required.

Each EDA standard is listed with its name, purpose, owning organization, points of contact for the standard or for ordering information.

1 Introduction

1.1 Charter

The EDA Standards Roadmap Workshop was initiated by the CAD Framework Initiative (CFI) through ARPA funding, Electronic Design Automation Companies (EDAC), and SEMATECH with participation by interested industry groups. The Workshop was specifically aimed at developing an industry-wide roadmap for development of design and test standards within the Electronic Design Automation (EDA) industry.

Three working groups were charged with identifying requirements, understanding the current standards environment, and developing a roadmap to improved standards across the next decade for their area of focus:

- EDA Interoperability and Integration Working Group (EII)
- Technology Libraries and Models Working Group (TLM)
- Design and Data Management Working Group (DDM).

The charter of these working groups is discussed below.

1.1.1 Identify the EDA Industry Requirements on EDA Systems

A primary goal was to identify the target requirements of the EDA industry on EDA Systems over the following timeframes:

- Immediate (Short) term (1995-1998)
- Near term (1999-2001)
- Long term (2002-2004 and beyond).

1.1.2 Review Status and Current Plans of Related Standards

With an understanding of the EDA industry requirements, develop a mapping to the relevant standards involved in supporting those requirements, and review the status and plans of current EDA standards that relate to those requirements.

1.1.3 Identify Standards Areas Requiring Improvement

Identify potential standards convergence opportunities, areas where new focus on existing standards work is needed, and areas where acceleration of planned standards work is required.

Clearly identify elements of the standards roadmap in the following categories:

- standards which should be *converged* in areas of overlap
- standards required which are *new*
- standards areas under development that should be *accelerated*

1.1.4 Determine How to Coexist and Migrate to Improved Standards

Identify the standards changes necessary to meet the requirements, as well as the appropriate coexistence and migration plans from the legacy standards to the improved EDA system standards structure.

1.1.5 Develop a Roadmap to Future Standards

Keeping in mind that the perspective of the Roadmap should be geared towards the world-wide community, the working groups should develop and deliver a roadmap of key action plans to support the requirements and itemize the standards-related work tasks required over time (using the definition of timeframe above). The roadmap should include task descriptions of each of the work items in the roadmap.

The task descriptions have different levels of precision depending on the timeframe for implementation. Those tasks required in the Immediate or Short Term (e.g., end of 1998 to coincide with next generation of CMOS .25um) will have more detail than tasks in later timeframes.

The roadmap for the short-term (immediate) timeframe should define the steps that the standards organizations must take to achieve the recommended converged state of standards to support the Technology Roadmaps. The roadmap steps will define the detailed implementation plan for items in the short term. Roadmap items that are targeted for the near or long term may be less defined.

1.1.6 Deliver Recommendations and Roadmap Contributions

CFI, EDAC and SEMATECH will assist the work groups and provide the lead for the final integration of the individual workgroup recommendations and contributions into an overall EDA Standards Roadmap. Tasks will include:

- Prioritize recommended actions
- Document the needed timeframe for implementation
- Seek approval and support of the Roadmap and its recommendations through the EDA Standards Industry Council.

1.1.7 EDA Standards Industry Council

The Industry Council includes individuals with the credentials and influence to support the Roadmap and promote industry adoption. The membership represents a broad range of geographic, academic, and government interests and reflects the major constituencies of the EDA industry. Industry Council members include:

- Robert Rozeboom, Texas Instruments, Incorporated (Chairman of the IC)
- Joseph Borel, SGS Thomson Microelectronics
- Ron Collett, Collett International
- Joseph Costello, Cadence Design Systems, Inc.
- John Darringer, IBM
- Aart deGeus, Synopsys, Inc.
- William Evans, AT&T
- Richard Goering, EE Times
- Andrew Graham, CAD Framework Initiative, Inc.
- Alain Hanover, EDAC and Viewlogic Systems, Inc.
- Randy Harr, ARPA
- Lambert van den Hoven, Philips Semiconductor
- Greg Ledenbach, SEMATECH
- Lance Mills, Hewlett-Packard
- L.J. Reed, Motorola
- Wally Rhines, Mentor Graphics, Inc.
- Gadi Singer, Intel Corporation
- Gary Smith, Dataquest
- Kinya Tabuchi, Mitsubishi Electric Corporation
- Hitoshi Yoshizawa, NEC.

1.2 Background

The initial input to the working groups included several documents. The requirements described in Chapters 4, 5, and 6 summarize the key requirements identified in these documents and other sources. The key documents include:

- National Technology Roadmap for Semiconductors¹ (NTRS)
- SRC White Paper²
- IPC OEM Requirements³

As indicated in the National Technology Roadmap for Semiconductors (NTRS, or commonly known as the SIA Roadmap), the U.S. semiconductor community faces new challenges as it moves towards design and manufacturing of chip feature sizes less than .50 microns. This is not unique to the U.S. and is in fact a global problem. A few of these challenges span the entire spectrum of technology including chip cores, chips, MCMs, and boards (PCAs/PCBs), and they require major industry initiatives to develop effective solutions. The NTRS Roadmap states that the magnitude of the challenges listed below demands the special attention of the semiconductor industry leadership.

1.2.1 Productivity Improvement

The NTRS Roadmap suggests that the semiconductor industry will require productivity gains greater than the historical 30% per-year, per-function cost reduction. Achieving projected densities and projected growth will require unprecedented industry cooperation and standardization through consensus. Standards will be required to enable cost-effective factories, and EDA standards are key to the success of future design teams struggling to achieve design productivity and density objectives.

1.2.2 Complexity Management

The years 2007-2010 will see maximum chip sizes increase to 350-800M million transistors for microprocessor designs, and 210M-430M gates⁴ for ASIC designs. Successful development of designs this size will:

- require very large design teams and enormous effort
- involve considerable design complexity
- result in a huge amount of design data to manage.

Maintaining control of these designs requires innovative design approaches and tools that support hierarchical and incremental design changes. EDA systems need to provide sophisticated design process and information management capabilities well beyond what is available today.

1. The National Technology Roadmap for Semiconductors, Semiconductor Industry Association, 1994
 2. Design Needs for the 21st Century: White Paper, Edited by Dr. James Freedman, VP of Research Integration, Semiconductor Research Corporation, 9/94
 3. The National Technology Roadmap for Electronic Interconnection, Working Draft, March 1995
 4. The National Technology Roadmap for Semiconductors, Table 2, p. 16, NTRS, 1994

To support the rapid evolution of design environments, new and innovative EDA solutions must be inserted into complex design processes. Widespread use of design re-use technology and complex intellectual property for data and tools, within and between companies, will be a major enabling technology for designs of such complexity.

EDA systems must be designed in ways that allow “technology insertion” of new EDA innovations as they become available. EDA standards that support an evolving design environment are imperative.

Note: 1.2.1 "Productivity Improvement" and 1.2.2 "Complexity Management" from the NTRS Roadmap are clearly contained within the scope of work for this EDA Industry Standards Roadmap; however, Items 1.2.3 "Advanced Technology Development" and 1.2.4 "Technology Development Funding" which follow, are not primarily within the scope of this Roadmap.

1.2.3 Advanced Technology Development

Funding for high cost and long-term research efforts has been reduced both in the U.S. and the rest of the world. The resulting gap in infrastructure must be filled by members of the semiconductor R&D community.

Improvements in software engineering are required; software applications are now fundamental to design, manufacturing, and business processes. Software development is the least perfected of engineering disciplines, and there are known software quality standards and practices from other software domains which could improve the quality of EDA software.

Increasingly, design and manufacturing of complex electronic systems requires a cultural change, from local optimization, to global optimization of technology solutions across multiple engineering disciplines. Changing industrial cultures is a formidable and time-consuming task.

1.2.4 Technology Development Funding

Meeting the challenges of the NTRS Roadmap will require an increased expenditure of resources on research and development of technology from the already heavy levels of today. The key challenge is to clearly define requirements and find funding strategies that cover all critical needs.

1.3 Scope

This section defines the scope of the EDA Industry Standards Roadmap. It includes strategic direction for key electronic design and test standards, with specific focus on the design system infrastructure, tools and design data, and key packaging technologies, across all key design phases.

The scope is focused on EDA, and is specifically focused on:

- EDA *systems and software* standardization (NOT standardization of electronics hardware)
- EDA and key *interfaces* to the EDA domain (NOT standardization of other CAD domains)

- EDA *Roadmap* for standards development (NOT actual development of the standards)
- EDA *Standards Roadmap* (NOT a design process or algorithm development roadmap)

1.3.1 Electronic Design and Test Standards Focus

Electronic design and test standards focus includes: infrastructure and tools, design and data management, design data representation, and technology libraries and models.

1.3.1.1 The Design System

This section summarizes the standards for the infrastructure surrounding and supporting electronic design and test systems and tools. It also covers all design and data management standards relating to the definition, execution, and management of the electronic design process.

- Infrastructure and Tools

This includes all standards dealing with the infrastructure surrounding and supporting electronic design and test systems and tools. The subject addresses platform operating systems, communications environment, user interfaces, tool encapsulation, inter-tool communication, and general EDA development environment. These standards promote innovation of new processes and methods for electronic design and test that can be easily inserted into the design environment (i.e., plug and play). Such standards operate at various levels to bind tools together into design flows and enable cost-effective multi-vendor flows without requiring tight integration of tools, and without impeding innovation within current or future tools.

- Design and Data Management

This area includes all design and data management standards areas relating to the definition, execution, and management of the electronic design process and its associated workflows and data across the enterprise. EDA design tool integration and general tool management standards are also discussed in this area.

1.3.1.2 The Design Information

This section addresses design and test design information (data) and the definition and re-use of design technology libraries and models.

- Design Data Representation

This area includes all standards areas relating to the definition and representation of electronic design and test design information (data) created and used by EDA design tools and/or the design team in the execution of the design process. Examples include such items as netlists, timing data, libraries, test benches, and many kinds of in-process data. The purpose of this area is to identify standards for EDA and point tool suppliers in order to support high quality and efficient information exchange and data sharing throughout the design process and across a geographically dispersed enterprise.

- Design and Technology Reuse

This area includes all standards relating to the definition and re-use of design technology libraries and models. Key library standards or standards requirements are identified, and focus areas and a roadmap for the next decade are described.

1.3.2 Technology Packages

The following technology and package types are addressed in this Roadmap:

- Technology Cores, Cells, and Macrocells (i.e., subsets of chips)
- Chips
- MCMs
- Boards (PCAs, PCBs, backplanes, subassemblies of various types).

1.3.3 Design Phases

The key design phases involved in design and test processes will be addressed, including:

- System Level Design (i.e., Architectural and High Level Design), and
- Detailed Design (i.e., Detailed Logic Design and Detailed Physical Design).

1.3.4 Key Electronic Design and Test Interfaces

The key interfaces involved between electronic design and test and other related disciplines will be addressed, including:

- Manufacturing Build Interface
- Manufacturing Test Interface
- Mechanical Design Interface
- Software Design Interface

While not addressed as part of the scope of this version of the Roadmap, it is recognized that Technology CAD (TCAD) is also an important key interface. TCAD should be addressed in future versions of the Roadmap.

This page was intentionally left blank.

2 Executive Summary

This chapter summarizes the findings of the working groups, and provides essential information from which the Industry Council review presentations were drawn. Highlights and key messages to the Industry Council are included for:

- recommended standards roadmaps for each of the key areas
- recommendations for standards convergence, acceleration, and new areas for development
- recommendations for a modernized standards development process.

2.1 Roadmaps

The working groups and the charter are briefly reviewed below, along with an overview of the categories of roadmaps developed, and then the roadmaps for each category.

2.1.1 Introduction to the Roadmap

The Roadmap is designed as a high level plan that enables the Electronic Design Automation (EDA) industry to converge on a common set of standards for the next decade. A summary of the charter and scope of this work follows.

The EDA Standards Roadmap Workshop was sponsored by the CAD Framework Initiative (CFI) through ARPA funding, Electronic Design Automation Companies (EDAC), and SEMATECH with participation by interested industry groups. The Workshop was specifically aimed at developing an industry-wide roadmap for development of design and test standards within EDA.

Three working groups were each charged with identifying requirements, understanding the current standards environment, and developing a roadmap to improve standards over the next decade in their area of focus. The three working groups were:

- EDA Interoperability and Integration Working Group (EII)
- Technology Libraries and Models Working Group (TLM)
- Design and Data Management Working Group (DDM).

The charter of these working groups was as follows:

- Identify the EDA Industry Requirements on EDA Systems over time
 - Immediate (Short) term (1995-1998)
 - Near term (1999-2001)
 - Long term (2002-2004 and Beyond).
- Review Status and Current Plans of Related Standards
- Determine How to Coexist and Migrate to Improved Standards

- Identify Standards Areas Requiring Improvement
 - Convergence of Standards in Areas of Overlap
 - New Focus on New Work
 - Areas Requiring Acceleration.
- Develop a Roadmap to the Future Standards.

2.1.2 Overview of Roadmap Categories

The information in this document and in this executive summary section is organized into categories as follows:

- Design System (Infrastructure and Tools)

This part of the executive summary highlights the key Roadmap items for standards related to the:

- Computing Environment and User Interface
- Design Tool Communication
- EDA System Extension Language
- EDA Standards-Based Software Development Environment
- Design and Data Management areas.

- Design Information (Design Data Representations)

This part of the executive summary highlights the key Roadmap items for standards in the following design information areas:

- Common topics across design information (such as incremental processing, hierarchical processing, and design object naming)
- Common topics across design steps (such as timing, simulation controls)
- System Level Design (i.e., architectural and high level design standards)
- Detailed Design (i.e., detailed logical and physical design standards)
- Design and Technology Re-use topics, including Technology Libraries and Models.

- Key Interfaces to Other Domains

In this section, the key Roadmap items relative to the key interfaces associated with the general area of engineering design and test are summarized. The interfaces to manufacturing build and test are discussed, as well as high-level roadmaps for mechanical design and software/hardware codesign.

2.1.3 Design System Roadmaps

This section addresses the specific roadmap for the design system category. Details on each of the roadmap items can be found in Chapters 4-6. In the figures, the following scheme is used (also see the legend on the chart for Converge/Accelerate/New):

- The priority of items is generally indicated by its placement in the Roadmap; i.e., items to the left are more urgent than items to the right
- Standards that are candidates to converge over time into new standards are indicated by the converge color (green); the arrows indicate to which standard they converge
- Standards which should be accelerated are indicated by the accelerate color (light red)
- New standards are indicated by the new color (dark red).

It should be noted that all of the new standards (red) are proposed to include an information model, a programming interface (PI), and a file format.

2.1.3 "Design System Roadmaps" addresses the Computing Environment, Extension Language, Standards-Based Development Environment, and Design and Data Management. Refer to Chapter 4 for additional details on these items.

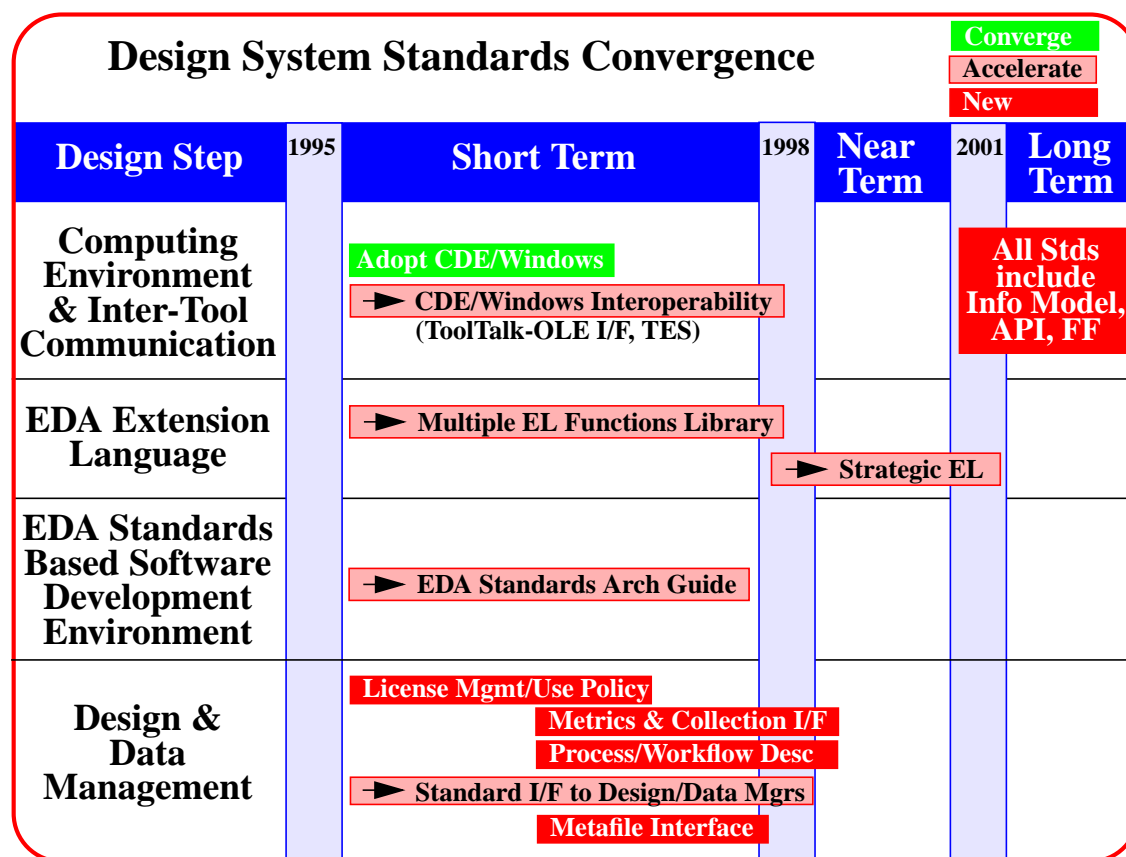


Figure 2.1—Design System Roadmap

2.1.4 Design Information Roadmaps

This section addresses the roadmaps that pertain to the standards for design data representation. The roadmap shown in Figure 2.2— "Design Information Roadmap" includes standards for the following areas:

- Common Topics Across Design Information, which include:
 - Incremental Processing
 - Hierarchical Processing
 - Design Object Naming
- Common Topics Across Design Steps, which include:
 - Timing Information
 - Simulation/Test Control
- System Level Design (i.e., architectural and high-level design standards)
- Detailed Design (i.e., detailed logical and physical design standards)
- Design and Technology Re-use topics, including Technology Libraries and Models.

Key Electronic Design and Test Interface Roadmaps, e.g., Manufacturing are also shown.

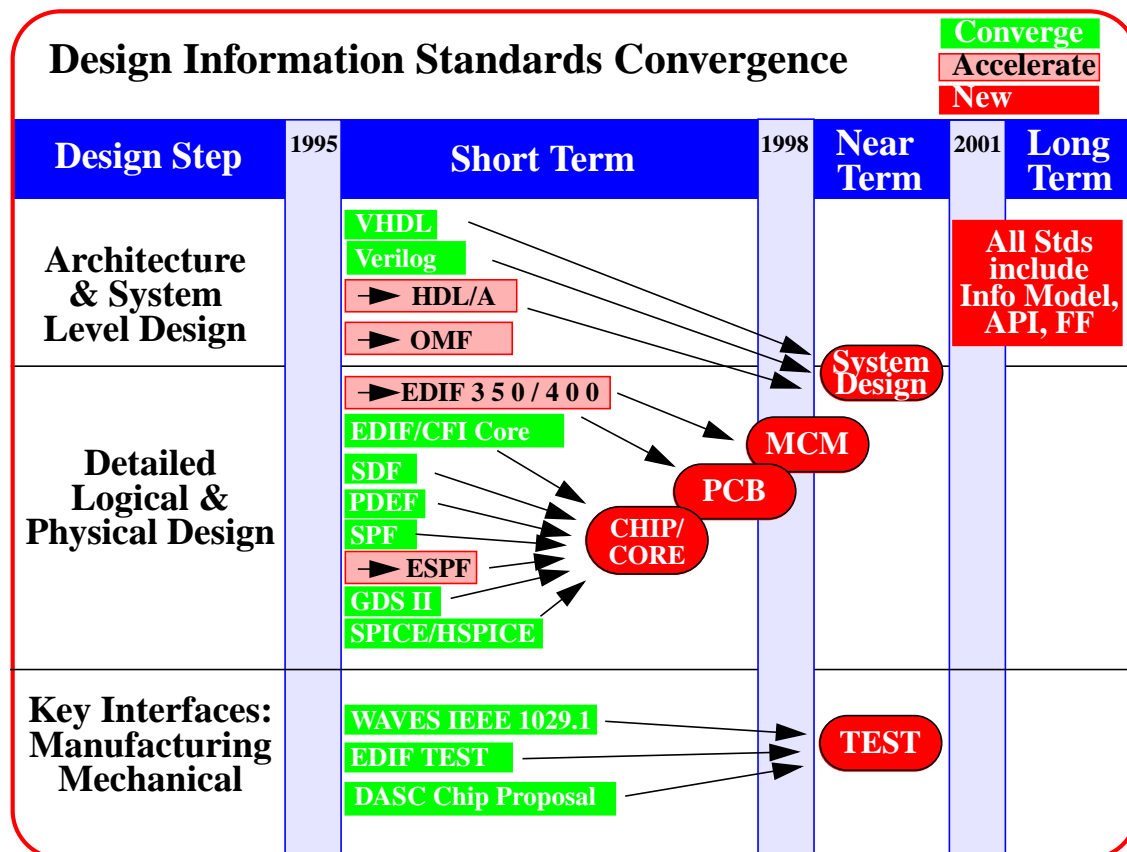


Figure 2.2—Design Information Roadmap

2.2 Recommendations

This section summarizes the key design data representation recommendations. Additional information providing backup and support for these recommendations is detailed in Chapters 5-6 of this book.

2.2.1 Key Roadmap Recommendations

The key design data representation recommendations of the EDA Industry Standards Roadmap are also highlighted in Figure 2.3— "The EDA Industry Standards Roadmap". Along the roadmap on the chart are the timeframes; short term (through 1998), near term (through 2001), and long term (2004 and beyond). The column entitled "Current Standards" lists key current standards in use by the EDA industry today, arranged in groups entitled "Systems Design" and "Detailed Design." Along the bottom of the roadmap, "Coexistence", "Migration", and "Evolution" characterize some of the key goals of the roadmap strategy. From left to right, there are a series of recommended EDA Industry Standards, which are explained below.

The following standards discussion includes references to information modeling which is further described in 2.3.2.1 "Technical Approach".

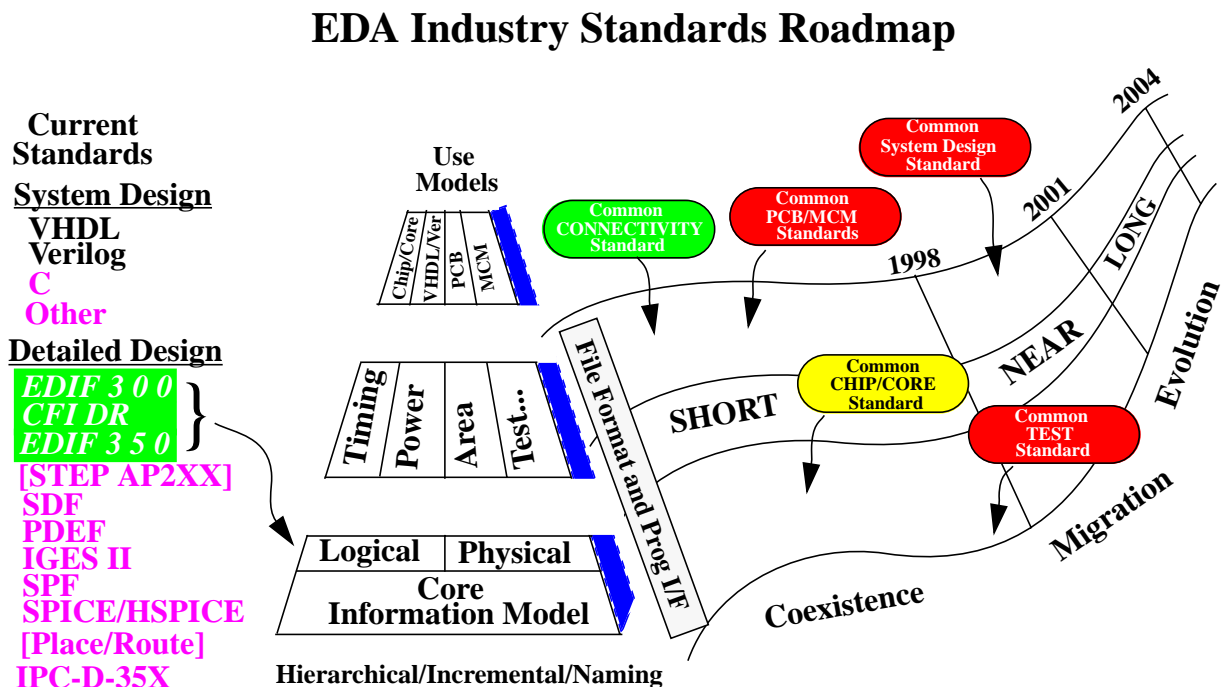


Figure 2.3—The EDA Industry Standards Roadmap

Common Connectivity Standard

In this phase, which is based on the common core information model from the EDIF - CFI DR converged model work, the framework is laid from which *all* the other design data representation standards developments evolve. This release establishes, for the first time, a file format (in this case an interchange language format based on EDIF 3 0 0), and a programming interface (based on CFI DR 1.X), that are based on the *same* information model. This work is also related to the international initiative to establish EDA interoperability with EDIF and CFI participating in a tiger team which began work in October 1993, under IEC TC93 (Design Automation), Working Group 1 (Interoperability). In addition, note the “Hierarchical, Incremental, Naming” at the base of Figure 2.3— “The EDA Industry Standards Roadmap”; this denotes the significance of providing support for the hierarchical and incremental processing of design data and for establishing common naming interchange conventions in *all* future standards development.

From this time forward, the idea is to add functionality to the base connectivity standard for various “use models”, based on design topics such as timing or package type (e.g., PCB or chip), or even system design models (based on VHDL or Verilog use models), including the necessary support for hierarchical and incremental processing, and for name mapping. All future standards releases must have a matched pair of file format and programming interface (PI). Refer to Section 2.3.2.1 “Technical Approach” where this approach to standards development is discussed in more detail.

Refer to 5.4.4.1 “Converged Industry Standard for Logical Connectivity” for additional information on this roadmap item.

Chips and Cores/Macrocells Standard

In this phase, use model standards for physical design data for chips and subsets of chips (cores or macrocells) are supported. A file format and PI are both available and are based on the extensions made to the common information model.

Over time, today’s de facto standards that contain chip/core information will converge into this industry-wide standard. For example, the planned SEMATECH/CFI effort on chip representation is committed to use this representation. The Roadmap says that the timing information that is today contained in SDF files could in the future be contained in a “chip/core” standards-compliant applications database. Because of the standards strategy of information modeling, followed by a file format (or language) and a PI, it is possible to coexist with legacy standards (e.g., the SDF) while migrating to a new standard (the chip/core standards). In order to meet the design requirements detailed in Chapter 5, it is imperative that vendor tools be developed as soon as possible to this new standard; however, because of the ability to coexist with legacy standards while migrating towards new standards, new tools that operate from the new standard can exchange information with legacy tools. EDA vendors can migrate to the new standard as their business situation dictates. At any point in time, the industry would have some vendor tools that operate on legacy standards and other tools that capitalize on the new standard, and the strategy recognizes the reality of this and supports it.

PCB and MCM Standards

In this phase, the physical information required to support board level packages including PCBs (printed circuit boards) and MCMs (multi-chip modules) is added to the common information model for connectivity developed in the previous step. There are two standards (or use models) released in this phase; one for boards and one for MCMs. A file format and programming interface are both available and based on the extensions made to the common information model as described above. This work should begin as soon as possible and the physical information should be initially based on the EDIF 3 5 0 for PCB (and eventually EDIF 4 0 0 for MCM) information models.

System Design Standard

In this phase, which is slated for consideration before 1998, a common information model and *standard for system level design* is developed and added to the information model base from the previous step. Because VHDL is significantly more comprehensive than Verilog for system design, VHDL could be a base from which to determine the information model for the initial system design standard base. Similarly, the information model developed from VHDL could be extended if necessary to completely cover the information model requirements for Verilog. Additional requirements to support any additional new system design language requirements can also be factored into the information model in a similar fashion. From this converged information model a system design standard with a PI can be developed with appropriate mappings to any HDL (e.g., VHDL and Verilog).

Using this strategy for VHDL, as an example, would allow the vision to be realized for a new VHDL use model standard; i.e., there is an information model from which a file format (i.e., the language, which in this example is VHDL and already defined) and a programming interface is developed. A new VHDL use-model standard could be released in the near term that includes both the VHDL information and a PI to access information in PI-compliant tools. Over the long term, the strategy states that for the file format, the system design standard should move to a common exchange file format that supports incremental processing.

It is important to note that in as much as the information modeling efforts between VHDL and Verilog overlap (i.e., the information being modeled is the same information), then the PI to access that information is *identical*. If, over time, as extensions are made to the system design information model (e.g., for analog support), then the PI for those extensions will be the same for both VHDL-based and Verilog-based customers, even though the file format representations (i.e., the languages) will remain different. It is anticipated that over time, more and more applications will migrate to the PI approach for data access in cases where data sharing is desired. It would be very desirable if the information modeling efforts described above were to be converged between VHDL and Verilog, but convergence is not required to achieve VHDL-based or Verilog-based use-model standards that have a PI as well as a language.

It should also be noted that this strategy for standards development supports the identification of areas where VHDL and Verilog have a direct mapping (e.g., the areas where the information model and PI are the same), and also those areas where the information models are different, and hence where their use model standards would be different. The creation of language translators would also be facilitated by this information modeling work; however, it must also be noted that because of the differences in VHDL and Verilog, translation between those lan-

guages is definitely not automatic and human intervention may well be required in such translation. The goal is to use the information modeling-based strategy, and not to require data translation to help us coexist with the existing legacy languages while we migrate towards an information and data sharing approach.

Defining an information model derived file format (designed for effective tool to tool data exchange of incremental design information, not designed to be human readable) would make it possible for any *specific* language dependency to be reduced over time. This strategy enables new HDL languages, including graphical representations (non-textual), to be directly supported by the system design standard and PI, and thus be less dependent on language form. Ultimately, it will be a combination of the users and the tool developers who will determine the rate of movement from the traditional system design language-based approach to more picture-based system design approaches that are to be supported by the system design standards. Again, this strategy for standards evolution is independent of that rate of change. Refer to Figure 4.2— "Open EDA Data Interoperability Architecture" for additional comments on coexisting with legacy languages and file formats.

Common Test Standard

In the area of test, it is also envisioned that a common test information model should be developed from which existing legacy test standards can be supported. A common test standard can be developed to enable design and test support software and hardware to meet emerging design and test requirements. In the same fashion as in the other standards, the test standard also would have a file format and PI available based on the common test information model.

Many commercial and de facto standards exist today. However, the data formats for these standards are largely incompatible with each other, making information exchange and data translation difficult. There are several efforts underway (IEEE ABBET and EDIF TEST committees are examples) to further refine and consolidate these standards and to identify areas requiring additional standards development.

A common test information model that supports the existing standards while enabling migration to a common test standard in the future is a necessary first step towards compatibility. The PAP-E (PDES Application Protocol -Electronics) program has developed a core information model for test and integrated diagnostics written in EXPRESS, which supports bindings to existing standards. This model is currently in review within ISO as proposed STEP AP211. It is also being used as the baseline model for defining EDIF TEST. Bindings currently exist to WAVES for test vectors and to EDIF 3 0 0 and 3 5 0 for product information. The core model is easily extensible to add bindings to other standards and technologies.

This Roadmap proposes that the PAP-E model be used as the starting point in defining a common test information model. As the PAP-E program has demonstrated, if the test interface is standardized via an information model it becomes less important to standardize on particular data formats (such as WAVES for test vectors) as long as bindings or mapping models exist between the data formats and the information model.

In the future it will be more important to enhance the existing standards and develop new standards for data formats with an eye toward supporting "lossless" mappings to the standard information model.

2.2.2 Coexistence and Migration Strategy

The Roadmap recommendations described above will be extremely difficult to implement without an effective method for coexisting with today's standards while attempting to migrate to a better standards-based future.

In order to support coexistence and migration, legacy standards such as SDF, PDEF, and other file formats must be supported, while the industry migrates to a more focused strategic standard such as a common PI and related file format(s)/languages. To support coexistence and migration, a family of translators must be developed and made available between each important legacy language (file format) and information model compliant data repositories. With this approach, only one certified translator for each language in each direction (i.e., one for import and one for export) would be needed.

This strategy also makes it possible to gracefully coexist with and migrate to new standards-based design data repositories that include *mixtures* of legacy data and data based upon the new standards roadmap. For example, tools that depend upon SDF files today can still operate without change, and new PI-based design data repositories can import/export SDF transparently through the PI (i.e., the PI knows where and how the data is to be stored or retrieved). Tools that are being rewritten (e.g., to support incremental or hierarchical design) can access that data directly via a PI or via a common standards-based file format.

To the extent that this standards development process (i.e., information modeling, file format and PI) approach is adopted by industry, it would enable legacy design languages and interchange file formats to eventually phase out as primary design representations. Coexistence with legacy formats would be supported by the translator set and new tool development would be done entirely to a new incremental file format and PI based on the information model. The PI approach also insulates the tool developers from the specific technology used to actually store and retrieve the information such as relational data base technology, object oriented technology, (e.g., OMG CORBA), etc. Client applications are also completely insulated from the specifics of the database implementation, and may be distributed across a LAN or WAN, through the use of the PI approach. As a result of this approach innovation is enabled, yet tools can be implemented in an open EDA environment.

2.2.3 Areas of Convergence

This section summarizes areas where ongoing standards work has considerable overlap. This overlap has been identified and the recommendations contained here suggest that certain standards be strategically converged. Table 2.1: "Areas of Recommended Standards Convergence" summarizes the recommended convergence of standards.

As indicated in Figure 2.3— "The EDA Industry Standards Roadmap", examples where standards efforts must be converged include:

- The convergence of EDIF 3.0.0 and CFI DR 1.X into an industry standard information model that encompasses both of them. In the figure, the base core information model and the common connectivity standard reflect this approach.

- The convergence of common design topics such as timing and physical parameters including parasitics, floorplanning and placement, and wiring information must be converged over time via extensions to the common information model described above. Over time, this could enable the potential phaseout of several current standards, including:

- SDF
- PDEF
- IGESII
- SPF
- SPICE/HSPICE (potential)
- various floorplanning, place, and wire file formats.

Note that the phaseout of the above standards was described as potential; this is because the strategy allows the phaseout to be gradual based upon EDA vendor development capabilities and user demand with the “coexist and migrate” strategy described above. The proposed strategy enables all EDA tools to continue with the file-based interface to design data for as long as they have value to users; but the real goal is to get to the PI. The key to achieving standards convergence on a programming interface is that the information model that the PI is based on must meet all requirements for data expression currently serviced by existing file-based interface standards.

- The convergence of standards for all types of packages and levels of design based on a common information model from which various “use models” for each package type are developed. Use models for boards (PCA/PCB), MCMs, one for chips/cores/macrocells, and a system design language are recommended.
- The convergence of standards related to the manufacturing test interface is also possible (but less studied and understood), and is based on the same strategy as described above for the design and build standards for packages. That strategy is to determine the information model requirements based on existing or legacy standards, and then develop a file format and PI that supports a converged information model. The convergence of standards related to the manufacturing test interface is possible using such a common information model to represent test information and by providing bindings or mapping models to commonly-used commercial and de facto standards. Additionally, product design information required by the test development process can be converged with the test information model via a mapping to the converged EDIF - CFI DR information model. As in the package standards convergence examples above, the test standards in use today should converge with potential phaseout of selected test-related standards. This convergence of test standards should cover:
 - STEP AP211
 - WAVES
 - EDIF Test
 - IEEE ABBET.

Table 2.1: Areas of Recommended Standards Convergence

Current Standard	Recommended Standards Convergence
EDIF	See 5.4.4 “Detailed Design Representation Standards”
CFI DR	See 5.4.4 “Detailed Design Representation Standards”
SDF	See 5.4.4 “Detailed Design Representation Standards”
PDEF	See 5.4.4 “Detailed Design Representation Standards”
IGES II	See 5.4.4 “Detailed Design Representation Standards”
SPF	See 5.4.4 “Detailed Design Representation Standards”
SPICE/HSPICE...	See 5.4.4 “Detailed Design Representation Standards”
STEP AP211	See 6.2 “Manufacturing Test Interface Standards”
WAVES	See 6.2 “Manufacturing Test Interface Standards”
EDIF Test	See 6.2 “Manufacturing Test Interface Standards”
IEEE ABBETT	See 6.2 “Manufacturing Test Interface Standards”
DCL	See 5.2 “Timing Information Standards”
IBIS	See 5.2 “Timing Information Standards”

2.2.4 Areas of Acceleration of Work

This section includes areas of ongoing standards work that require an accelerated rate of development to meet current and future design requirements. A boost in funding and/or resources or EDA vendor focus may be required to accelerate the work of development or adoption of a standard. Refer to Table 2.2: "Areas of Recommended Standards Acceleration" for a list of areas where standards work should be accelerated.

Table 2.2: Areas of Recommended Standards Acceleration

Current Work	Recommended Standards Acceleration
OMF	5.3.4.3 “Standard Interfaces to Design Analysis Tools”
ToolTalk	4.9.4.1 “Intertool Communications: Adopt ToolTalk...”, others
Tool Management	4.9.4.3 “Establish EDA Industry License Manager/Use Policy
DCL Project	5.2.1.4.3 “Complete Delay Project and Extend Beyond ASICs”

2.2.5 Areas Where New Standards Work is Required

This section highlights areas where either gaps in standards exist or no standards exist at all. New standards are required to meet critical design requirements, now and in the future. Refer to Table 2.3: "Areas of Recommended New Standards Work" for a list of areas where new standards work is required.

Table 2.3: Areas of Recommended New Standards Work

Related Standards	Recommended New Standards Work
ToolTalk, OLE 2	4.2.4.2 "Provide Windows Interoperability for ToolTalk"
VHDL, Verilog, C	5.3.4.1 "Standards for System Level Design"
Library Standards	5.5.4 "Roadmap - Design and Technology Re-Use"

2.2.6 Areas Where Additional Roadmap Work is Required

This section describes areas that require additional study to complete a detailed roadmap. In these areas, the working groups ran out of time and/or resources, and it was determined that additional follow-on work is required before conclusions can be reached. Refer to Table 2.4: "Areas of Recommended Additional Roadmap Development" for a list of areas where additional roadmap development work is required.

Table 2.4: Areas of Recommended Additional Roadmap Development

Related Standards	Recommended Additional Roadmap Development Work
Library Standards	5.5.4 "Roadmap - Design and Technology Re-Use"
Board Standards	5.4.4.2 "Converged Industry Standard for Board Packages"
Data Management	Chapter 4 Design Data, Resource, and Release Management
Manufacturing Test	6.2 Manufacturing Test Interface
Mechanical Interface	6.3 Mechanical Design Interface
Software Interface	6.4 Software Design Interface (Hardware/Software Co-Design)

2.3 The Standards Development Process

This section describes the current standards environment and describes a recommended technical approach to standards development and a standards management process.

2.3.1 Current Standards Development Environment

The most widely-used standards support tool interoperability through the use of “standard” interface file formats. Standards efforts working on the definition of these file formats tend to be somewhat disjointed, and to a degree, competitive. This requires industry players to either support all of these different standards or pick the one(s) they feel will most likely satisfy their customers. To complicate matters, the standards tend to differ not only in format, but also in content and in the basic structure of their information models. For example, the basic information model for EDIF 3 0 0 electrical connectivity is different from that of CFI DR 1.X or STEP AP2XX. This leads to a further need for translation software to convert between these different standards, which is costly, short lived, and prone to errors.

To meet the challenges of the future, there are a number of standards related needs that must get into focus:

- EDA standards efforts must be harmonized in line with a single “roadmap” (as described in this document). Some harmonization efforts have started, such as between CFI and EDIF, but this is “ad hoc” and without long-term targets or goals. There is considerable confusion and frustration across the EDA industry, the ASIC suppliers, and the end-user design community. A roadmap-driven standards effort will offer a strong and stabilizing base from which to build a more effective long-term set of standards.
- The time period between development of a standard and when that standard ships as part of a commercial product is much too long. A considerable reduction in this time should be a mandatory requirement in reforming the standards development process. The time required for standard identification, definition, industry acceptance, and certifiably accurate implementation needs to be addressed, and requires much closer cooperation between EDA industry participants.

2.3.2 Standards Development Process Recommendations

This section describes the technical vision of standards development and proposes a management model for standardization of future developments.

2.3.2.1 Technical Approach

This section describes the vision and the engineering of standards development from a technical viewpoint and the strategy for coexistence and migration towards this engineering approach from today’s situation.

In general, for a given EDA standards area that needs significant work (and where it makes sense), the information engineering process of building an EDA information standard should include:

- Definition of an information model (in EXPRESS)

An information model for the EDA domain should initially focus on defining the semantics of the domain. This includes the construction of a formal specification of the information in the model (objects, attributes, and relationships). To eliminate ambiguity, this model should fully cover every aspect of the application wherever possible. This information model must be designed to be extensible in order to meet emerging and unforeseen requirements. There must be traceability between the information model, the subsequent use models (or application models) built upon it, the programming interface (PI), and exchange file formats. The work of EDIF and CFI to develop the converged information model addresses such requirements.

The engineering process used to create, test, and verify this information model should gain leverage from ongoing industry methodologies in information model development being used by EDIF, CFI, SEMATECH, STEP, RASSP, and others. The technology of information-modeling engineering itself is undergoing standardization and should be monitored and exploited as appropriate.

- A programming interface (built from the information model)

A software programming interface (PI) is designed to support an access method for data *sharing* and direct data access (i.e., no data exchange required). This approach can also be used for high performance data *transfer* between PI-compliant tools. When implemented in a programming interface appropriately, this approach is inherently hierarchical and incremental in nature. This approach can also be used to provide a strategic PI interface between client and provider applications that enables support of legacy file formats, while coexisting and migrating to more strategic forms of design data and PI access. This means that a client application can request design data through a PI interface and be relatively independent from the actual source of the data (e.g., an SDF or a new converged standard that contains the timing information).

- A file format (built from the same information model)

The file format approach is designed to support the *exchange/archival* of design information where appropriate, such as when passing data between two companies or between a development site and a manufacturing site (where the programming interface concept of data sharing may not apply). Clearly, file formats also provide a base from which to *translate* design data where translation is an effective mechanism.

It is desirable that a binary computer readable (not human readable) and incremental “stream” file format that is based on the information model be developed to suit data archiving and incremental data exchange between information model compliant applications. Such a stream file format could minimize or eliminate the necessity to continue extending source languages in favor of the stream file to meet exchange requirements.

A key element of the strategy is that *both* the file format and the programming interface should be developed and *delivered simultaneously* as new releases of the standard evolve.

To better convey this concept, refer to Figure 2.4— "Vision of Standards Development".

Vision of Standards Development

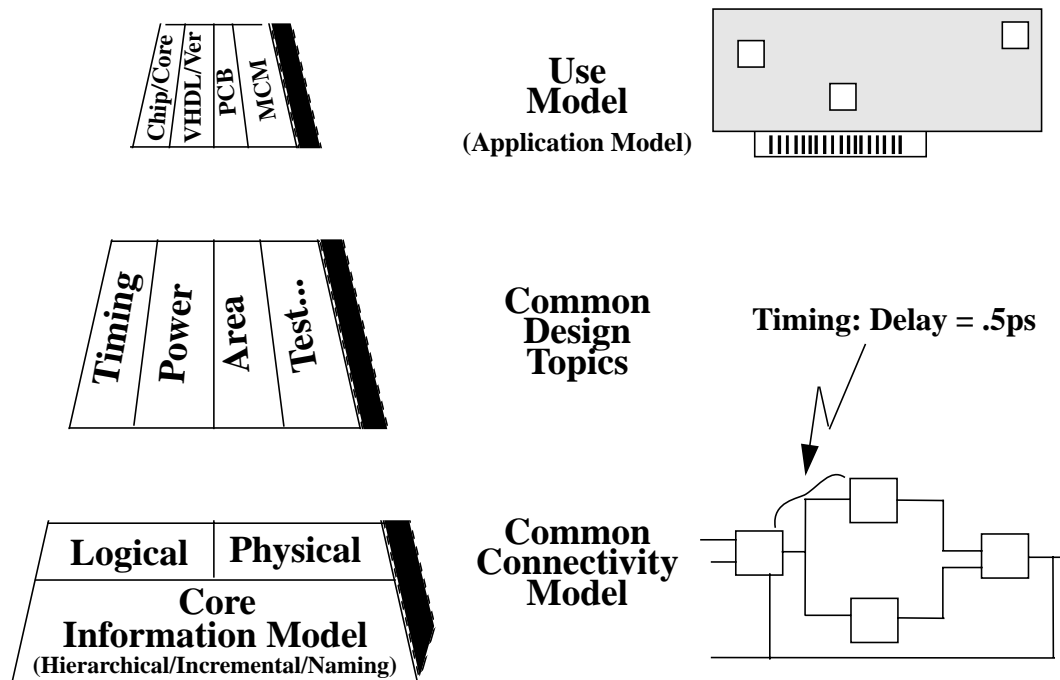


Figure 2.4—Vision of Standards Development

The concept is as follows:

- The Common Connectivity Model

A common core information model is developed (in EXPRESS) for fundamental design connectivity (which is the same connectivity model for all package types). That common connectivity model must also ensure and record the mapping and relationship of all logical and physical information about the entities being connected.

- The Design Topics Model

In this layer, the idea of various design topics such as timing, power, area, and test information are modeled (also in EXPRESS). Each additional design topic for which a standard is required must be added in this fashion. For example, a timing value for a net is shown as .5ps; this timing information would be “annotated” to the base connectivity model as information regarding the net timing information between a specific output net of one block and a specific input pin of another block.

- The “Use Model” Standards

In addition to the information modeled in the previous layers, certain additional information is required to support “use models” (or application models) for specific types of packages and the release to manufacturing of design information for those package types. For example, there is information that is unique to board (PCA/PCB) packages, MCMs, and also for chips/cores/Macrocells. The example in the diagram is one of a PCB. In general, there is a subset of the information in the common connectivity model, the design topics layer, and the physical package type that when considered collectively make up the use model for that type of package.

Similarly, there could also be strategically a use model for system design (e.g., a VHDL use model). By adding system design domain information to the base connectivity model and creating appropriate design topics or extensions thereof, the information model could be extended to cover the system design domain as well as the “package type” use models described above.

Any test of compliance to a “use model” standard actually implies compliance to proper processing of the information *per the use model involved*. In other words, for a given use model, a subset of the information model is important. For example, for a chip standard, chip pad information is important, while for PCB’s, edge connector information is important. Checking an application for compliance to a chip standard involves checking for proper processing of the chip pad information, and not edge connector information.

The vision here is one where the design data information model evolves from a common core information model, upon which specific design topics can extend that core model (e.g., timing), and from which various “use models” (e.g., a package type such as “chip”) can be further developed. Similarly, a use model for system level design can be developed. A “standard” can be developed for each of these use models, to which EDA applications can be certified or checked for compliance to the use model standard.

2.3.2.2 Standards Management Model

This section discusses the existing standards situation, and proposes a management strategy supporting the cooperative and focused development and adoption of future standards.

In the current arena, many standards organizations have their own standards that they develop and promote. Because of this, the EDA marketplace often experiences significant and unnecessary overlap, and competition in selected standards areas. This problem must be addressed by the Industry Council.

It is recommended that the Industry Council take a very active role as described in this roadmap document with respect to exercising some control over the standards that the council supports. Specifically, the Industry Council should act as the EDA industry “funnel” for standards efforts being proposed for development and/or use by the industry. All new standards proposals should be explicitly endorsed by the Industry Council before user groups or EDA vendors adopt such standards. Standards efforts that have not been explicitly endorsed by the council should be rejected. In this fashion, standards groups and other efforts to create standards will quickly learn to use the Industry Council as the appropriate forum for positioning and direction-setting for EDA standards activities.

It is also recommended that the Industry Council come to a common agreement that *all* EDA standards, and particularly design data representation standards, be developed under the technical process described in this document and when ready (i.e., accepted by industry), be approved by the Industry Council to be submitted to a *single* standards body (e.g., IEEE in the U.S., CENELEC in Europe, etc.) for formal standardization process and life-cycle management. All standards which are submitted to such standards bodies that are not endorsed by the council should NOT be supported by the EDA industry at large.

2.3.2.3 Pilot Programs and Prototype Development

Any new standard or significant new release of a standard should have a prototype implementation using the draft level of the standard to form a candidate standard.

Such prototyping or pilot programs should be targeted at real-world design problems using real EDA tools, modified to support the draft standards involved. It is via these pilot programs that key issues with the draft standards get identified and resolved before balloting as a new standard. Coexistence with legacy standards as well as support for migrating to the new standards must be demonstrated.

2.3.2.4 Test Case Development/Management

Test cases for large chip designs and for PCBs and MCMs should be gathered (or constructed) and made publicly available to facilitate and promote use in the testing of new tools developed by vendors and universities.

Developing test cases would be helpful in creating tools whose function can be demonstrated in real-world EDA environments and be more production ready. These test cases must be kept current and a plan must be put in place to ensure that the test cases do not become outdated. Such test cases could also be part of a certification process or any important benchmark process for standards.

2.3.2.5 Standards Toolkit and Conformance/Certification Plan

Any new standard or significant new release of a standard should have a conformance or certification plan and test suite, along with appropriate development toolkit software and reference implementations to promote uniform application of the standard, thus promoting true interoperability among tools implementing the standard. Where applicable, certification should be accomplished by and through the use of an appropriate test suite to certify the implementation.

2.3.2.6 Productization Support

Industry support of and by the EDA vendors to develop new or upgraded tools to the standards must be provided throughout the above process, from concept through prototyping, through to actual released products that support the standards involved. A strong and concerted effort (including the necessary resources and funding) is required to move products to the marketplace that are based on or depend upon new standards.

This page was intentionally left blank.

3 Electronic Design and Test Environment

In this chapter, environmental topics that are relevant to the effective design of complex electronic systems are discussed. Emerging paradigm shifts and many of the pressures on design and CAD integrator teams are identified and discussed. The detailed implications on standards that were a direct result of these topics are further discussed in Chapters 4-6.

3.1 Emerging Paradigm Shifts

Significant paradigm shifts within the electronic design community are dramatically changing the face of design and practices and are imposing new requirements on EDA tools and systems. These are discussed in this section.

3.1.1 Innovation in Systems Level Design (Architecture and High Level Design)

Many of the design process and tool innovations will occur at the front end of the design process. New tools are emerging to support the specification and analysis of design architectures, including performance evaluations, design trade-off analysis, hardware software co-design, and estimations of various factors such as life-cycle costs. Standards are needed which will minimize the resource and schedule impacts of inserting these new tools into the design process.

3.1.2 Innovation in Design Process Management

As the size and complexity of chips and electronic systems increases, the size of design teams required to develop them increases. The complexity of managing design flows and iterative design changes will also increase dramatically. Design teams may also be dispersed geographically. It is anticipated that designer productivity and increased codesign requirements will drive innovations in the area of concurrent design. Standards are needed to support the definition, execution, and monitoring of the integrity of the customer's design methodology.

3.1.3 Increased Codesign Across Design Disciplines

Technology evolution, e.g., deep submicron semiconductor technology, is forcing major discontinuities in traditional design methods. The complexity and scale of integration, as well as significant cost of design errors, promotes a re-evaluation of design practice and an increase in "co-design" or collaborative (and concurrent) design early in the design process and which spans multiple design disciplines. Parallel efforts in disciplines such as design for test, mechanical, software, hardware/software, and electrical design are being driven earlier and earlier into the design process. Standards are required to enable these design disciplines to communicate with each other to maximize concurrency and improve design quality in the design and manufacturing processes.

3.1.4 New Architectural and Integration Concepts

The massive size and complexity of chips is encouraging the integration and increasing re-use of chip cores and other design objects coming from different application domains (e.g., telecommunications, computing, and biomedical) into one design. This will demand whole new architectural concepts and may include much more programmability to turn general architectures into application-specific products. This in turn will drive the development of new EDA tools, design processes and methodologies, and libraries to support these concepts. Standards are needed to support design and technology re-use to enable design productivity in these design scenarios where multiple design domains are very closely integrated.

3.1.5 Changing Business Practices

The business of developing electronics-based systems is undergoing significant changes, which are expected to continue during the next decade. Some of these changes include:

- Engineering design, and EDA environments for doing design, are now subjected to stringent Return-on-Investment (ROI) analysis.
- ROI analysis fuels trends such as outsourcing of selected phases of product development including the actual functional design, physical design, or manufacturing of a design object.
- Re-use is a major design consideration.
- Much more manufacturing knowledge (build, test, cost, yield, etc.) must be brought early into the design process.

These and other changing design practices will lead to different business practices such as those for intellectual property (e.g., chip cores, design processes, and others).

In addition, the existing model for EDA tool support (maintenance) is typically based on a hardware service model. This approach does not generate adequate revenue for EDA research and development, and at the same time, EDA customers do not perceive enough value for the high cost of maintenance. This approach also fuels the changing business practices in the EDA tool development environment. New approaches and standards are needed to support different ways of doing business between EDA tool suppliers and their customers.

3.1.6 Pay-Per-View for Design Tools

The geographically-distributed product development environment necessary in the next decade will enable other new EDA design tool marketing and distribution approaches where tools are “rented” for use in design projects. This will enable small enterprises or even individuals to be able to afford previously unreachable design tools. For example, designs could be securely shipped to “design centers” for physical design (common today) or tools could be licensed to designers on a pay-per-view basis to do any design activity. Standards are needed which will minimize the resource and schedule impacts of inserting these new practices into the design process, such as consistent license management policies for tools and reusable design objects.

3.1.7 Object Oriented Software Development

The emergence of object oriented software development paradigms such as that from OMG and CORBA show promise for improving software development productivity over the near and long term. While it is too early to adopt specific standard interfaces to these technologies, the Roadmap recognizes the potential need to migrate to such standards because they demonstrate significant returns for the EDA industry. Also, strategies for coexistence with current technologies (e.g., ToolTalk and others) while we migrate to standards such as OMG/CORBA will need to be developed.

3.2 Pressures on Designers and CAD Integrators

Electronic engineering environments range from small enterprises using loosely connected tools to very large, complex, distributed, and heterogeneous multi-company (virtual) enterprises. This range of design environments contends with significant pressures on design and CAD integrator teams as shown below.

3.2.1 Exploit Multiple EDA Operating Environments

In the workstation environment, UNIX is currently the mainstay for EDA tools with growing interest in Microsoft Windows NT. On the PC platform, Microsoft Windows is dominant, with a limited chance of any other significant players emerging in that operating system environment. Today, many companies have design flows that exploit more than one of these operating environments. This is expected to continue. There is a growing urgency for a single design flow to effectively use both operating environments as if they were one environment. Standards are needed to enable EDA tools to interoperate and to communicate across multiple operating environments.

3.2.2 Use Diverse Databases and Formats

Many design groups do not use Product Data Management (PDM) systems today, relying on a loose network of tools fed by various forms of netlist files. As designs get larger and re-use becomes more prevalent this must change. There are many different PDM systems in use across the EDA industry today. Product data management systems exploit relational databases as well as object-oriented database technology. This area of database technology is a rapidly emerging technology that is expected to continue growing. The challenge for EDA systems is to be able to exploit and capitalize on the best data management technologies as they emerge. Standards are needed to enable enterprise-wide data management across geographically dispersed and diverse operating systems in this changing database environment.

3.2.3 Use Tools from Multiple Tool Vendors

There is a critical need for new EDA tools that help designers meet goals for minimum time-to-market on their products. Designer productivity is a major issue now, and the pressure on designer productivity will only increase as technology moves into deep submicron. Commercial EDA companies will continue to strive to develop new tools and capabilities that meet

productivity needs. It is clear, however, that the “best of class” tools will not all come from one vendor, especially when the time constraints of the NTRS Roadmap is considered. Tools from multiple vendors need to behave as if they come from one vendor because CAD integrators and designers will want to use the best practice tool in their design flows, since it helps establish their competitive advantage. Standards are clearly needed to support the insertion of new tool technology into the design process as it becomes available.

3.2.4 Enforce Design Methodologies and Process Management

As designs become larger and more complex, the numbers of designers will also increase in size. In order to keep track of the state of various elements of the design (e.g., chip cores), hierarchical and incremental approaches will be required. Managing the state of the design process for each of these design elements in a concurrent engineering environment, will be a major challenge. It is expected that improvements in process and workflow management will be major issues in the next decade as the complexity of designs increases. Standards to support multiple design managers and multiple design processes across the entire enterprise are required.

3.2.5 Reduce (or Maintain) Cycle Time

Design teams are faced with continuing pressure to minimize the cycle time associated with the development cycle, in the face of dramatically increased complexity of the electronics. While the design cycle time is currently getting longer, there is a strong demand to increase the “circuits per day” productivity metrics even as design complexity grows. This improvement in productivity must be met without sacrificing design quality. Standards to support the management of increased design complexity (e.g. timing constraints) and increased design and technology re-use can help in this area.

3.2.6 Reduce Design Costs

As always, there are pressures to minimize design costs while maintaining or improving quality. Major contributors to reduced development costs include reduction of design schedules, and production of a design that is “correct” the first time into (and out of) manufacturing. Standards which assist in the management of the design process and maintain the integrity of the design as it evolves are critical in this area, along with improved EDA tools.

3.2.7 Maximize Return-on-Investment (Price/Performance)

CAD integrators and design groups are well aware that their EDA systems design environment is a costly factor in the total cost of developing a product. At the same time, the product can not be developed without significant investments in design technology. The goal is to minimize product development time while maintaining maximum price/performance of physical assets such as workstations, PCs, operating system and communications software, and EDA software from multiple vendors. Measuring cost-of-ownership and return-on-investment for design technology is becoming a common objective in many design groups. Design groups are feeling increased pressure to run themselves like a business, with investment in design

technology being a significant portion of their operating costs.

These pressures on designers and on the CAD Integrators that support them impart several key requirements on the EDA system and on the design information contained within the EDA system.

The EDA design system environment and the associated EDA domain data standards are both key elements of the design environment. This environment creates requirements in major categories such as the Design System Environment, and the Design Information Environment. Additional environmental topics within each category are discussed in the subsequent chapters.

3.2.8 Design for Quality

The electronics industry is placing increasing focus on product quality (6-Sigma) and the effect of quality on customer acceptance, development, and field support costs. To achieve the desired quality levels, manufacturing build and test processes and the customer product environment must be considered concurrent with the original engineering design of the product.

The pressure on product design teams and on EDA tool suppliers to improve product quality will continue to increase. Improved quality of EDA tools could also dramatically improve product design cycle time and provide significant cost savings to EDA, semiconductor, and electronic design and manufacturing companies.

There are software industry standards and practices on quality such as the Waterfall Development Model, Spiral Lifecycle Model, Maturity Model from the Software Engineering Institute, and ISO 9000 software product standards which should be researched and adopted where relevant.

This page was intentionally left blank.

4 The Design System (Infrastructure and Tools)

This chapter includes the computing environment and user interface, design tool communication, extension language, software development environment, and several design and data management areas. These topics are primarily domain-independent topics but with an EDA focus.

4.1 Computing Environment and User Interface

The computing environment of the next decade will include UNIX and Microsoft Windows operating system environments, and potentially some others. Each of these environments have certain implications as described briefly below.

4.1.1 Current Environment - Computing Environment and User Interface

Two main platforms are in widespread use for EDA applications today: desktop and high-end server engineering workstations running variants of UNIX; and PC-type desktop machines running variants of Microsoft Windows. The most complex chip designs tend to be done using UNIX applications while PCs with Windows are used for selected PCB designs, FPGA and Programmable IC designs, and some low-end ASIC design (as applicable for selected portions of the design process).

The most popular platforms for EDA software use today include the UNIX workstation. The favored computing environments include:

- SunSoft SunOS, migrating towards Solaris 2.X and follow-ons
- Hewlett Packard HP-UX
- IBM AIX
- Digital UNIX (aka DEC OSF/1).

UNIX running on engineering workstations will continue to be a significant platform in many design groups for the next decade. This is due to many factors, including legacy investments in hardware, EDA software, designer training and familiarity, development of design processes and methods, and continued development of the workstation environment by manufacturers to offer competitive price/performance, particularly for very large computer servers for high-end design applications, such as large custom chip designs. All major vendors are members of the COSE alliance and are or will be very shortly, compliant with CDE (Common Desktop Environment), that will be the standard graphical user interface and window management environment.

In the PC world, Microsoft's Windows (Windows 95, Windows NT, etc.) is a clear leader. Also, OLE 2 and Microsoft-endorsed CAD standards continue to lay groundwork for future EDA applications on these platforms. It is anticipated that Microsoft Windows and its variants will be the dominant PC operating environment over the next decade. Some leading analysts

have indicated that Windows will emerge as a more widely-significant EDA operating environment, though a significant investment in EDA applications redevelopment for that environment will be required. The availability of back-porting kits for Windows-based applications to run on UNIX workstations, as well as on PC platforms based on Windows, will continue to promote a merger of the EDA marketplace into one large market divided into overlapping subsets rather than the more distinct and separate markets of today.

Apple OS 7 is the environment for legacy Apple hardware, with OS 7.5 on current Apple PowerPC platforms. Current announced industry strategies from Apple indicate PowerPC platforms will be migrating towards the “Copland” operating system environment, which is Apple’s answer to Windows 95. There are no major EDA applications on this platform today. IBM’s OS/2 Warp, or Taligent, or any follow-ons are not envisioned as contenders in the EDA marketplace at this time. While none of these operating environments are expected to be major players in the EDA world in the next decade, they should be monitored.

Generally, the direction of the computing environment is towards being cross-platform; i.e., the hardware (workstation or PC) is becoming less relevant. It is anticipated that over the next decade UNIX will be available on PC platforms and that Windows will be available on workstations.

In addition, given the prediction that future designs will involve groups of geographically separated designers, it is expected that networked computing will become the new paradigm for the design environment in the not-too-distant future, allowing groups of people to communicate and cooperate on a single design. This technology is in its formative stages with Lotus and Novell as the current leaders and Microsoft promising its own support based on Windows 95. It is too early to tell at this time who will set the norm for this type of computing environment, but the requirements are relatively clear.

4.1.2 Requirements - Computing Environment and User Interface

Below are listed the key requirements in this area.

4.1.2.1 Consistent User Operating Environment

In the geographically-dispersed, multiple operating environment described above, the important point is that, regardless of the suppliers, the environment, as seen by the user, must provide the same capabilities and “look-and-feel”, independent of the underlying hardware and software.

Investments in EDA design system platforms, software, environments, methods, and user training need to be optimized as these elements evolve, and are replaced, or updated. Legacy investments will be optimized if the learning curve for designers moving to new platforms or OS environments is minimized. Since the coexistence of UNIX and Windows based platforms and applications is now occurring, there is both a strong need, and real opportunity to identify standards for common operating environments, and for commercial industry to realize and adopt these standards.

In order to maximize the effectiveness of design teams the operating environment and user interface of the design system and its tools play an important role. Standards need to be established that minimize the learning curve for designers moving to new platforms or new operat-

ing system environments whether they are UNIX-based or Windows-based. In many ways, the UNIX and Windows environments have migrated to the point that they are similar enough to each other that a user of one environment can quickly adapt to the other. Only recently has the UNIX world had the ability to exploit a common user interface that is similar to that of Windows (and Mac), with the advent of the Common Desktop Environment (CDE). There is a requirement for EDA vendors to adopt these operating environment standards in order for designers to maximize the EDA and design technology resources, regardless of platform and operating environment.

4.1.2.2 Consistent EDA Environment

In addition to a general domain-independent user interface, EDA users, CAD integrators, and EDA vendors would benefit from a consistent vendor-independent EDA environment that includes “Consistent User Operating Environment” above and also provides a “standard EDA toolbox” that addresses the requirements listed in the rest of this chapter. These standards-based EDA tools include such things as the EDA Message Dictionary for Tool Communication, Tool Encapsulation support, extension language support, and potentially an EDA software development environment (e.g., for promotion of EDA standards-based tool development in the university environment). Each of these examples of EDA appliances must be “certified compliant” to the standards it supports to gain customer acceptance as a totally compliant EDA user (or developer) environment.

4.1.3 Recommendations - Computing Environment and User Interface

From a computing environment and user interface perspective the EII working group feels that there is “enough” consistency between the user interfaces of CDE, Windows, and Macintosh that there are no major unresolved issues in this area.

The working group recognizes *both* UNIX and Windows as players in the next decade and without attempting to explicitly pick a winner or emphasize one over the other, recommends that a “coexistence and migration” strategy be developed and supported between them.

This recommendation should be periodically reviewed as new environments evolve (such as IBM’s Taligent or Apple’s Copland).

The next point of focus should be to provide the EDA user community and possibly the EDA developer community with a standards-compliant environment for EDA software development. Thus, a set of compliant EDA tools should be made readily available to the EDA industry quickly to establish a solid design system computing environment and graphical user interface foundation based upon key EDA standards from which future innovations can evolve.

4.1.4 Roadmap - Computing Environment and User Interface

This section provides recommendations for the graphical user interface and EDA development tools.

4.1.4.1 UNIX: Adopt CDE as Graphical User Interface

The recommendation on this item is to adopt CDE on UNIX platforms as the standard for graphical user interface desktop management. Interoperability with the Windows environment is an emerging need. See “4.2.4.2 "Provide Windows Interoperability with ToolTalk",” for additional information related to tool interoperability between UNIX and Windows applications.

This item should be adopted as high priority immediately (i.e., in the short term).

4.1.4.2 Windows: Adopt Windows Graphical User Interface

The recommendation for this item is to adopt Windows on PC (and/or workstation) platforms where justified. This situation should be monitored over the next decade as Windows environments evolve.

This item should be adopted as high priority immediately (i.e., in the short term).

4.1.4.3 Standard Base Operating Environment for EDA

This roadmap item is to define, develop, and deliver a base set of certified EDA tools in a package that meets requirements for domain-independent and selected EDA domain needs in the area of tool communication and encapsulation, EDA system extension language support, and potentially-selected EDA domain data translators. This tool suite should also provide UNIX and Windows interoperability in a standard way. The EDA development environment is another candidate suite of tools.

This item should be pursued as medium priority for the short term.

4.1.4.4 Establish Formal Path for EDA Industry Requirements on OS Providers

A specific process and path for identification and management of EDA industry requirements should be developed to provide focus to the OS providers on the key needs of the EDA industry. The recommendation here is to assign a neutral EDA industry representative to interface to the OS providers and to have all OS requirements be funneled through that body.

This item should be adopted as high priority immediately (i.e., in the short term).

4.2 Design Tool Communication

This section documents the major design tool communications issues and provides recommendations for inter-tool communication standards.

4.2.1 Current Environment - Design Tool Communication

Electronic design systems are rapidly evolving to geographically-dispersed, network-based environments. Major design tool improvements are needed to meet the challenges of the future. The designers are struggling to manage the design process with the massive amounts of data involved. Approaches are needed that enable tools to perform cooperative computing via collaborative efforts on design problems. It must be possible for tools to interact in intelligent ways via inter-tool communication and messaging, reaching across the worldwide network.

4.2.2 Requirements - Design Tool Communication

4.2.2.1 Standards for Inter-Tool Communications

There is a requirement to reduce design cycle time in key design loops by maximizing inter-tool communication and interoperability performance. Through the use of dynamic inter-tool communication between active applications of incremental design changes, as opposed to sequential file translation and transfer of entire design sections, the overall cycle time for key design process loops could be significantly reduced. The required participation of the design engineer in these design loops can be greatly reduced to improve designer productivity. Standard inter-tool communications technologies are required to support this communication.

In addition such a technology would enable EDA design tools to also achieve a higher level of independence from other EDA support technology such as Product Data Management (PDM) technology (if messages were used to interface to the database provider being used by a client application in an enterprise) and from the environment in general. Tools such as workflow or process managers, or any EDA tool, could then communicate with PDM systems in a fashion that enables the insertion of new technology whenever it becomes available. Related information on the topic of inter-tool communication are in section 4.9 "Design Tool Management", 4.6 "Design Management", and 4.7 "Design Data Management".

4.2.3 Recommendations - Design Tool Communication

The ToolTalk messaging facility has already been endorsed by CFI as the standard inter-tool messaging mechanism. CFI has developed an EDA Message Dictionary (draft) standard that was designed to meet many of the above requirements, thus the recommendation for ToolTalk with the appropriate EDA Message Dictionary extensions. This approach can meet requirements in this category in the UNIX environment.

In addition, extensions to ToolTalk must be provided to meet requirements for inter-tool communication between tools that are running in the Windows environment and those in the UNIX environment. Such interoperability would best be provided by the operating environment suppliers (i.e., the UNIX or Windows providers). In the absence of that solution, alternative suppliers of such functionality could potentially be found.

Additionally, as mentioned in 3.1.7 "Object Oriented Software Development", support for object oriented software development standards based on OMG and CORBA may be required. While it is too early to adopt specific standard interfaces to these technologies, the roadmap recognizes the potential need to migrate to such standards as they demonstrate significant returns for the EDA industry. Strategies for coexistence with current technologies (e.g., ToolTalk and others) while we migrate to OMG/CORBA will need to be developed.

4.2.4 Roadmap - Design Tool Communication

This section includes the recommendations for UNIX and Windows platforms.

4.2.4.1 UNIX: Adopt ToolTalk as the Standard ITC Mechanism

The recommendation on this item is to adopt ToolTalk on UNIX platforms as the standard for inter-tool communication.

This item should be adopted as high priority immediately (i.e., in the short term).

4.2.4.2 Provide Windows Interoperability with ToolTalk

The recommendation on this item is to enable inter-tool communication between Windows-based software and UNIX applications. The recommendation is that a Windows OLE to ToolTalk interface be developed to enable that communication. Further, it is recommended that Windows-based vendor developments adopt OLE as the base for their ITC mechanism. These actions would tend to support consistent ITC mechanisms across multi-platform operating environments and further support other roadmap recommendations for design and tool management.

This item is a “new standard,” that needs funding, and it should be pursued with medium priority immediately (i.e., in the short term).

4.3 EDA System Extension Language

This section includes the recommendations for EDA systems and toolsets.

4.3.1 Current Environment - EDA System Extension Language

Extension languages are an integral part of most current EDA systems and toolsets. They provide designers and CAD integrators with an important mechanism to “glue” tools, often from multiple sources, together into design workflows and processes.

Popular extension languages in use today including SKILL, AMPLE, and Scheme-based extension languages such as the CFI Extension Language (EL). In addition, scripting languages such as PERL (and TK/Tcl) are frequently used to extend EDA systems in the UNIX environment.

The result today is the unavoidably wide variety of proprietary and non-proprietary extension language scripts found in most real EDA system environments. The multiplicity of languages, and the proprietary nature of some, means that major EDA design systems functionality that uses extension languages may be difficult and expensive to maintain, to migrate to other toolsets, and to migrate across releases of the EDA tools and design system. Such code is often treated as “throwaway” code, and as a result, is poorly written and documented. This further reduces its long-term effectiveness in providing designer (and CAD integrator) productivity. There is currently no plan to migrate to a more strategic language.

There is no extension language in the Windows environment today, at least not formally a language designed for that purpose.

4.3.2 Requirements - EDA System Extension Language

This section includes the recommendations for an EDA System Extension Language (EL) and EL Functions Library.

4.3.2.1 Standard EDA System Extension Language (EL) Functions Library

A standard EL Functions Library, at least for the UNIX environment, that provides access to EDA design data and design system objects (via a set of standard reusable EL Functions Library, including access to EDA standards-based facilities for design data representation, inter-tool communication, tool encapsulation, user interface constructs, etc.), is required. This library of reusable software objects offers functions and capabilities that:

- Can be used by the application developer or by the CAD integrator/user
- Provides a consistent graphical user interface where needed
- Offers a consistent set of application controls to CAD integrators/users as well as application developers
- Is accessible from all popular extension languages; today this includes CFI EL (Scheme) and PERL.

4.3.2.2 Standard EDA System Extension Language (EL)

In the short (immediate) term, in order to support legacy extension language contributions, multiple extension languages may need to be supported in the UNIX environment. However, over the near and long term, a single strategic EDA extension language (EL) standard is ultimately required to enable the creation of portable, reusable, migratable, well-crafted, and documented extensions by designers and CAD integrators.

This EL must have the following characteristics:

- Allow ease-of-use and learning
- Support interpretive (rapid easy development) and compiled (efficient to use) forms of execution
- Provide access to EDA design data and design system objects (via a set of standard reusable EL Functions Library, including access to EDA standards-based facilities for design data representation, inter-tool communication, tool encapsulation, user interface constructs, etc.) as indicated above
- Allow binding of C/C++ code
- Be portable across all vendor tools and operating platforms (and where tools support access to the EL Library, they operate to give identical results)
- Support dynamic module/library loading (to isolate the EL library functionality from the EL itself)
- Have other characteristics as defined in 4.1 "Computing Environment and User Interface".

This EL may be required in the Windows environment, although those requirements are less clear.

4.3.3 Recommendations - EDA System Extension Language

Given the volume of legacy EDA code developed in the above languages, a strategy must be devised to support legacy code while attempting to migrate to a more strategic solution.

The CFI Extension Language (based on Scheme, which is an IEEE standard) has been previously recommended for support by all EDA vendors and tools, as a standard extension language, with limited success. PERL has emerged as perhaps a leading contender for the most popular extension language. Additional study to determine which of these candidates (or others) should be selected as the strategic EL for the EDA industry, is recommended. There is no mandatory reason why *one* EL must be selected in the short and near term.

In addition, there is no extension language for the Windows environment

What is perhaps more important, is that access to EDA data, messaging, and other objects be adequately supported, in a standard way, via an EL Library. This library should contain the functions necessary to support the writing of the necessary glue code independent of the EL used. Additional requirements as they emerge should be added to the EL library rather than leading to the creation of new or extended proprietary extension languages.

In addition, extensions to enable PERL to access the EL Library facilities via an API should be supported, along with tools to assist users with eventual migration to the strategic EL.

EDA vendors should develop translation and migration tools to assist in migration to the standard EL at some future major release. EDA vendors should stop extending their proprietary languages, while continuing to maintain them.

4.3.4 Roadmap - EDA System Extension Language

This section provides recommendations for a standardized EDA system extension language and functions library.

4.3.4.1 Provide Multiple EL Access to EL Functions Library

Appropriate API bindings to support multiple EL language access to a common EL Functions Library need to be developed. Languages for which this access is required include CFI EL and PERL.

The CFI EL Library meets some of the documented requirements for an EDA extension language library, and is recommended as the base for the strategic industry standard EL Library. The library of functions that make up the existing CFI EL Library should be supported and extended, to meet new requirements (e.g., DR object access, TES object access, User Interface access, etc. per 4.3.2.1 "Standard EDA System Extension Language (EL) Functions Library" for EDA Extension Language).

This item should be adopted as high priority immediately (i.e., in the short term).

4.3.4.2 Select and Standardize on Strategic EL Language

Additional work is recommended to build consensus on the best strategic extension language. By focusing on the EL library access method from the short-term extension languages (i.e., CFI EL Scheme and PERL), the priority of this task can be moved to later in the short term (i.e., medium priority).

Also, once a strategic EL language has been selected and adopted by the EDA industry, migration tools should be developed by EDA vendors to enable easier migration to the strategic EL when the user chooses to make that move.

4.4 EDA Standards Based Software Development Environment

This section provides the requirements and recommendations for the EDA standards based development environment.

4.4.1 Current Environment - Standards Based Development Environment

The computing environment described in Chapter 2 "Executive Summary", is driving the need for EDA tool developers including EDA vendors, proprietary development in user companies, and in university and other research environments to develop their products to operate on multiple hardware and software environments, including both UNIX and Windows. At the same time, development of applications should be done to a number of various other EDA standards, as documented in this roadmap. Porting or simultaneous development of standards-based tools compatible with all popular environments will be important in ensuring timely availability of tools in customer environments and in the cost of insertion into commercial use at reduced cost.

Developments in the university and other research environments is now done in an “ad hoc” fashion with no direction from the EDA industry to enable innovative new tool development and easier adoption by the industry.

4.4.2 Requirements - Standards Based Development Environment

University, research, commercial, and in-house developers of EDA tools need standards to assist in reducing the cost of development, and improve the standards-based quality of EDA software solutions for design problems.

There is a requirement for a “standards-based development environment,” i.e., development with a focus on appropriate EDA standards, in which to develop tools that can be released to the multiple customer environments in use today and in the future. Such a standard development environment would be of *major* value to developers everywhere.

A good example of the benefits of a standard development environment would be in the University research environment, where innovations produced using the standard development environment would more likely be “adoptable” in the industry; i.e., because the tool was developed to popular EDA standards, and can run on popular EDA operating platforms.

4.4.3 Recommendations - Standards Based Development Environment

An EDA industry-wide standards architecture document that reflects the recommendations and roadmap steps outlined in this document should be developed.

4.4.4 Roadmap - Standards Based Development Environment

4.4.4.1 Open EDA Standards Architecture Guidelines

A task team must be put in place to develop an Open EDA Standards Architecture Guidelines document that defines and documents the guidelines for development of EDA applications to better support “plug and play” in the supported EDA computing environment as prescribed by this roadmap. This document would address the concepts, architecture, and relevant EDA standards (e.g., computing environments, inter-tool communication, tool encapsulation, design data representation, etc.) which should be used to develop tools that can be easily adopted by industry upon successful completion of the application. This document should become the key reference document for standards-based EDA software development.

It is specifically recommended that the Industry Council take management ownership of this architecture, and sponsor the development of these guidelines with the assistance of appropriate EDA industry experts, for review and endorsement by the council.

A preview of a possible architecture is given in Figure 4.1— "Open EDA Enterprise Architecture". The enterprise in this case is defined to mean all of the people, processes, hardware, software tools, metadata, libraries, and product data associated with the development of a product. There are several important considerations of note in this architecture, including:

- The use of inter-tool communications in:
 - tool launching (e.g., by CDE, and by workflow managers, and by frameworks)
 - tool to tool communications such as in the interface to data management for such actions as “check-in” or “check-out” of design objects from or to the design workspace
 - tool to tool communications for the handling of incremental design changes as opposed to reprocessing the entire design.
- The use of Tool Encapsulation files (TES - see 4.9 "Design Tool Management") for *all* EDA system related tools. The TES files enable the automatic encapsulation and integration of tools in a consistent manner independent of the way the tool will be launched (e.g., from CDE, from the workflow manager(s), or from a framework).
- The concept that all tools expect their data to be in the local “design workspace,” and that workflow managers and framework managers should make use of the ITC data management message set to ensure that this is true, so that legacy tools can operate in such environments without change. In addition, tools which would like to capitalize on interfacing directly with the data management services layer *may* do so with the same ITC messages, but they are not required to do so.
- Access to the EDA System Extension Language Functions Library from any of the standard EDA Extension Languages (see section 4.3 "EDA System Extension Language") supports access to EDA system objects (i.e., ITC messages, design objects in the design workspace, TES files, graphics support, etc.). This facility makes it easy to extend the system using a standards-based extension language and functions library. Such functionality must be accessible from script and/or command line launching as well as from graphical user interfaces.

This item should be adopted as medium priority immediately (i.e., in the short term).

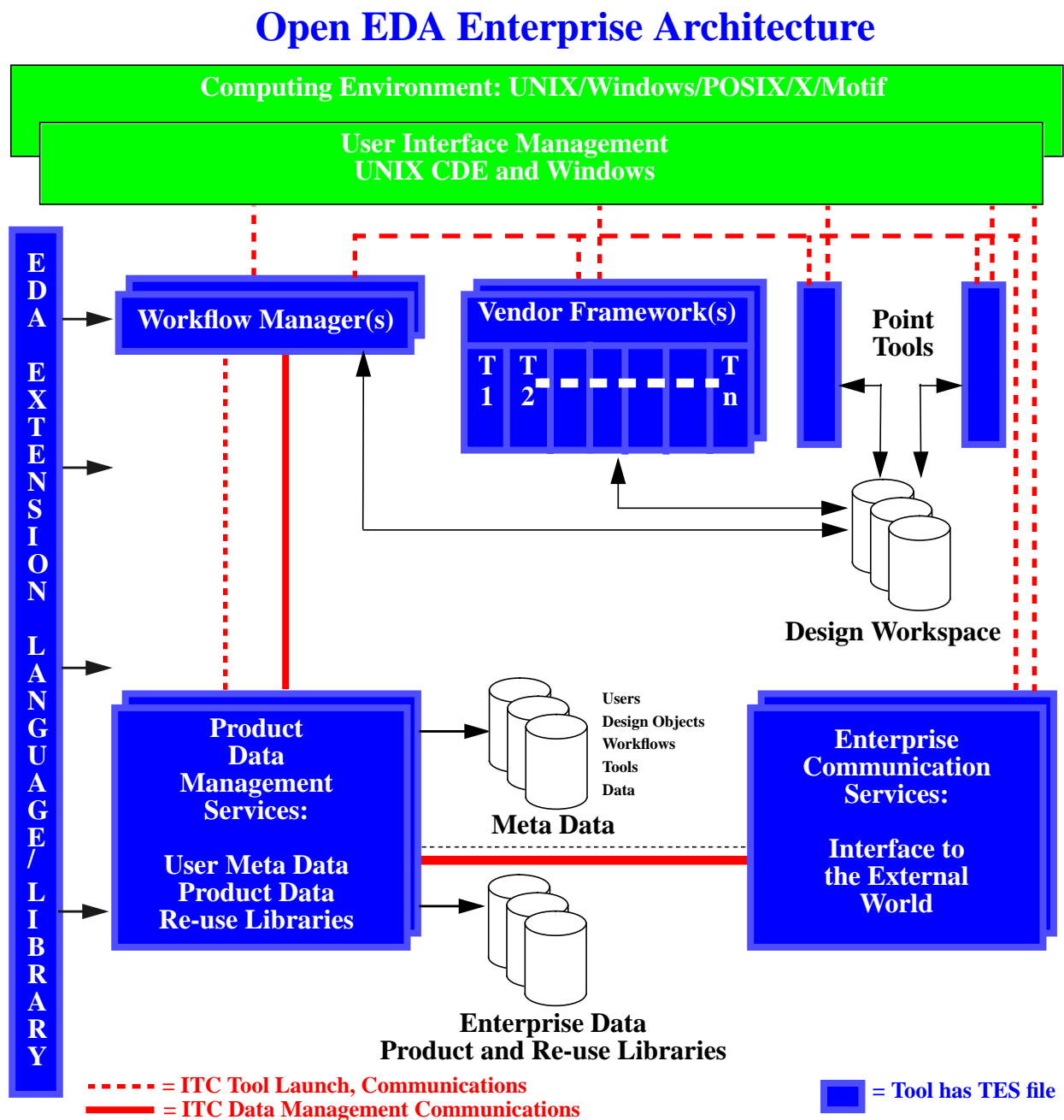


Figure 4.1—Open EDA Enterprise Architecture

Another important part of the Open EDA Standards Architecture is the design data standards and the data interoperability architecture which supports those standards in a way that coexistence and migration is supported. As indicated in Figure 4.2— "Open EDA Data Interoperability Architecture", a family of information model compliant client applications can be developed for each important legacy language or file format, as well as to vendor proprietary

databases, which can interface to the industry standard (e.g., the chip design representation standard). These client applications exist for both directions (i.e., import and export).

Open EDA Data Interoperability Architecture

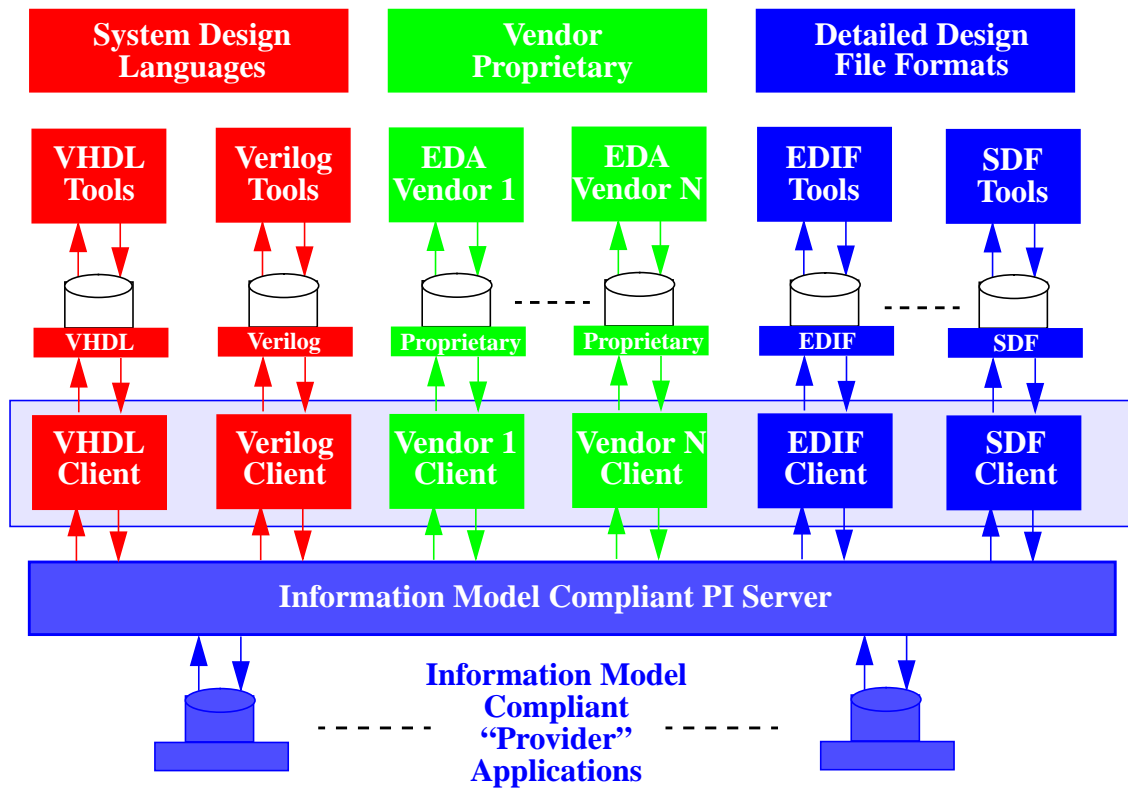


Figure 4.2—Open EDA Data Interoperability Architecture

4.5 Intellectual Property Protection for Design Data Objects

This section addresses intellectual property protection with respect to design data objects. The issue of license management for design tools is discussed in 4.9 "Design Tool Management".

4.5.1 Current Environment - Design Data Object Protection

In the case of asset protection and license management for design data objects, there are *no* standards.

4.5.2 Requirements - Design Data Object Protection

4.5.2.1 Standards for Design Data Object Asset Protection

There are requirements for standards for license management that support protection of intellectual property assets for design data objects. There is some overlap in the requirements to protect design data object assets and the requirements to protect design tool assets. The need for asset protection spans a variety of design data objects, including re-use items such as designs for re-use and various technology rules. Each of these examples (and many more) demand an effective solution to protect intellectual property. It must be possible for this information to be electronically distributed over a geographically dispersed network.

4.5.3 Recommendations - Design Data Object Protection

While this topic is important to the EDA industry (both developers and users), it is also viewed as a topic that is largely beyond the scope of EDA and more of the scope of the computing environment in general. Solutions to this problem are ultimately expected to come from the OS providers.

4.5.4 Roadmap - Design Data Object Protection

While this topic is important to the EDA industry, it is believed that the solution must ultimately come from the OS providers. A more strategic solution is anticipated in the short or near term because it is extremely important to commerce on the internet for industry at large, and not just for EDA. Therefore, it is recommended that the EDA industry drive to (and not implement an EDA-unique solution for) that enterprise-wide platform independent solution from the OS providers, and that the process described in 4.1.4.4 "Establish Formal Path for EDA Industry Requirements on OS Providers" be used in driving for that solution.

4.6 Design Management

Design management (or workflow management) encompasses automated support to aid or enforce a prescribed design methodology across a set of design objects, design steps, and design tools. Elements of design management support include:

- Formal definition of a design methodology (i.e., the design process or workflow)
- Completeness checking of design steps and control or assistance in determining the next allowed process step(s)
- Association of design data types and design tools for each design step
- Auditing of the design process to assure valid design hand-off
- Design status or state accounting.

Design projects involving large design teams or large numbers of design steps and tools are finding it increasingly necessary to incorporate design management automation to assure consistency and accuracy in the overall design process. While often true that design methodologies are loosely defined and enforced, there is a growing interest in formality due to the increased complexities of semiconductor design, the number of designers involved with a design entity, and the necessary interaction across multiple design domains and geographic locations.

4.6.1 Current Environment - Design Management

Although there is a diversity of design environments within the industry today, the general norm is a rather loose definition of the overall business process. The EDA design environment tends to be captured within scripts that define the tools required for the various design tasks written in extension languages such as SKILL, AMPL, PERL, Scheme, and TK/Tcl. The overall business process tends to be documented on paper and to use manual procedures to control the flow of design and information between people and organizations.

Design management (also called workflow management) tools are commercially available from EDA suppliers and other commercial companies. Each has particular functional strengths and capabilities in terms of system platforms supported, ease of use, ease of integration of tools, checking, reporting, etc. Some are better adapted to a specific suite of tools supplied by a given EDA company marketing the design manager, while others are more generic and provide support for EDA Mechanical and Manufacturing engineering design.

Many potential customers of such systems foresee the need to use multiple design managers within their overall design system (unique requirements of specific engineering design domains, local preference at different geographic locations, etc.) while others are exploring internal proprietary solutions as a result of their diverse needs. Research in this field is exploring the addition of artificial knowledge-based decision management capability that may someday provide automation to the design trade-off decisions engineering must make in today's complex designs.

In the current environment or state of the art for design management systems it is clear that:

- The formal encoded definition of a complete design methodology is a complex, labor intensive task and once completed, it is not likely to be redone just to reformat it
- Design methodologies themselves are customer and design-specific and cannot therefore be standardized (only the design process specification method can be standardized)
- Customers desire the ability to use multiple design managers and to change design managers with minimum impact should a “better” one become available (i.e., customer choice of “best practice”)
- Many potential customers of these products require the ability to move design information from one design manager to another as the design progresses. Large enterprises are at various stages of recognizing the need for formal design management systems. Some, such as in the RASSP program, are exploring the personalization and use of commercially available offerings, while others are doing internal research and development of proprietary solutions. Most design teams interviewed, however, admit the need for formal design management systems to meet future design problems and objectives.

4.6.2 Requirements - Design Management

The importance of automated design management is a function of the complexity of the design process (number of different process steps or number of design elements, for example), the number of engineers or engineering teams involved in the design, and the required interactions between the different design domains required for successful design.

Based on the following technology trends, the need for automation to aid in design management will steadily increase:

- Design complexity will increase
The number of design process and checking steps that must be successfully enacted to complete a design will increase.
- Design team size (i.e., the number of engineers) involved with a single design entity will increase
- Design interactions across multiple design domains will increase

Interactions between design engineering and manufacturing engineering in order to design for manufacturability will increase in importance as more sophisticated packaging evolves as will interactions between other domains such as electrical and mechanical design for PCAs, electrical and software design, etc. These lead to the following requirements.

4.6.2.1 Design Methodology Process Description Standard

The creation of a complete design methodology process description is a large and difficult task. Diverse groups within a single design enterprise need to use the same design process, but may need to use (or change to) different design management systems. It is imperative that a standard for representing this important design data be developed and adopted to enable design methodology to be portable across compliant design managers. Further, the overall business process may demand the use of multiple design management systems for a single

business process and potentially even multiple design management tools working on the same process.

Therefore, it is necessary that a standard for design process definitions be developed to meet the requirements for a portable and sharable process description, which would support the need for multiple design managers within an enterprise and the ability to change design manager products without the cost of re-coding the formal design methodology. This interchange standard must not impose restrictions on or create standard methods for entry and editing of the customer methodology.

4.6.2.2 Standard Interface between Design Tools and Design Manager

Potential customers of design management systems require the ability to encapsulate and execute design tools of choice (from multiple companies or locally developed) into the chosen design manager without the need to modify the tools.

Therefore, it is necessary to develop either a standard interface between the design management tools and the CAD tools to support tool activation and deactivation and that interface must be independent of and impose no specific requirements on the CAD tools.

Because business processes may cross multiple hardware-operating system platforms, the standards developed must apply equally well across UNIX, Windows, and other popular platforms.

4.6.3 Recommendations - Design Management

There are a number of industry organizations who are or have been involved with design management standards including CFI, OMG, and the WfMC. Government-funded programs such as RASSP and NIIP have placed a high degree of importance on design management systems and are encouraging the promotion of standards to meet the stated requirements.

Use of ITC message passing technology and standard messages is recommended as the standard interface between design managers and tools. Therefore, it is recommended that a set of ITC messages be developed and adopted by industry to meet the functional requirements for design management including tool launch, tool completion reporting, metrics gathering, etc.

4.6.4 Roadmap - Design Management

Efforts to develop standards for workflow managed environments and a portable design methodology process description should be accelerated.

4.6.4.1 Design Methodology Process Description Standard

It is imperative that a standard for representing design methodology process information be developed and adopted to enable design methodologies to be portable and sharable across compliant design managers to meet the above requirements.

For development of the methodology definition standard, it is recommended that the Industry Council commission an industry organization to define the requirements for this capability and issue an RFT to industry in order to build this standard off an existing industry-proven base. The final result of this activity should then be offered to a formal standards body (such as IEEE) for life-cycle support.

This item must be addressed with medium priority (i.e., in the short term).

4.6.4.2 Standard Interface between Design Tools and Design Manager

A standard ITC-based messaging interface between the design management tools and the CAD tools must be developed to meet the above requirements.

It is recommended that the Industry Council commission CFI to work with the above organizations to develop and promote a comprehensive set of ITC messages to meet the functional needs of design management and that CFI (or the X-Open consortium) provide a mechanism for life-cycle support of these messages. It is further recommended that OMG be commissioned for a parallel set of class objects.

This item must be addressed with high priority (i.e., in the short term).

4.7 Design Data Management

Management of design data is an essential practice for electronic systems design for all package types and across all system level design and detailed design phases. Management of design data (files and databases) necessarily includes the following elements:

- Data ownership and access authority information
- Managing concurrent data access
- Managing different levels (versions, iterations, etc.) of design data
- Design configuration (collecting all required data at the correct level for a design)

With increasing design sizes, projected increases in the number of designers involved with a design entity, concurrent design, and globally-dispersed design teams, the need for more formal and automated methods for data management (and using multiple data management sub-systems) is expected to increase in importance and provide a growing opportunity for commercial data management products.

4.7.1 Current Environment - Design Data Management

While commercial data management products are available from both EDA vendors and other commercial companies, they are often incorporated with proprietary solutions or other commercial products. This seems to be necessary in order to cover the total scope of data management requirements across large enterprises in particular. While all of the data management related commercial products offer strengths in one aspect or another, it is most often the case that no single product yet meets the total set of needs foreseen by the design community.

On the other hand, all of the commercial data management products offer a basic set of services which are essential to data management. Thus, there is a great deal of functional overlap between competing products and certain value-added features within each.

The current environment causes a number of problems to the potential customers and may actually be limiting the growth of commercial purchase vs. proprietary development. The specific problems in the current environment are:

- No standards for common data management functions

There are no agreed to standardized terms for the basic service functions of data management systems. The definition of commonly-used terms such as check-in, check-out, open, purge, update, submit, merge, retrieve, release, promote, version, level, workspace, etc., is open to interpretation, as is the function performed when any of these data management terms are called out for execution. This makes it very difficult for potential customers of these products to evaluate the central features of a data management product, let alone the merits of the value added features because they often cannot get an accurate understanding of the base on which they are built. Further, the complexity of integrating commercial solutions together or with internal solutions is compounded by these “semantic” differences, often to a point where an internally developed solution can be more robust and economical.

- No standards for metadata representation

Often, in large enterprises, there is a need to move data from one data management system to another as a design progresses between engineering domains or departments or geographical locations. This requires that the metadata (data records used by the design management system to manage the design data) needs to be extracted from one design data management system and reformatted or even rewritten for another. This mapping needs to be done in a fashion that guarantees no loss of information about the design; however, with conflicting and ambiguous terminology used between data management systems, this task is far more complicated than simple data reformatting. Further, the completeness and accuracy of such translation is suspect, again causing potential customers to favor internal solutions.

Design data management is an extremely complex problem when one considers hundreds or thousands of design files, at different levels and versions, and which must be correctly interrelated to assure accurate manufacture, concurrent use of design data by multiple people and organizations, and distribution across tens or hundreds of computers. Further, these complexities are growing in proportion to advancements in electronic technology. That being the case, design groups may well desire the ability to purchase design data management solutions from professionals well versed in the technology as opposed to internal investment into ad hoc solutions. However, the ability to do this is severely hampered by the lack of basic consistency standards as described above.

Making matters even worse are the number of disjoint consortium or government-funded efforts that are seeking to rectify some of these problems by “standardizing” selected elements of design data management, including RASSP, NIIP, STEP, WfMC, and others. These disjoint activities will compound the problem even further for the EDA industry as they will, no doubt, each create their own set of standards which will be different from the other efforts.

4.7.2 Requirements - Design Data Management

Below are the key requirements for the design data management category.

4.7.2.1 Standard Interface between Design Tools and the Data Management Subsystem

Any solution for design data management must be available as a design service that is independent of the approach to implementing design methodology, the EDA applications, and design files being used. Customers must be able to apply the chosen design data management solution on their desired design system of tools and must be able to do so without imposing restrictions on those tools and without the requirement for changes to those tools.

The CAD integrator must have the ability to call out design data management services from the user environment being used. This includes design methodology workflow managed environments as well as more manual design flow management using extension languages such as script file languages (as described in Figure 4.1— "Open EDA Enterprise Architecture"). In addition, it must be possible for EDA tools to choose to integrate such services without being required to do so. This ability must be available in such a way so as to allow for the customer to have the ability to choose or change the design management system as he/she would for any design tool.

In order to support the above, it is necessary to agree to a standard definition of the meaning and action of common design data management services. For each of these, an ITC Message should be defined and adopted to support the invocation of these services independent of the chosen management system and to support replacement of the design data management system without necessary change to the design environment or tools.

4.7.2.2 Design Data Management Metafile Interchange Standard

In order to support both the ability to incorporate multiple design data management systems within an enterprise and the ability to substitute different design data manager(s), a standard must be developed and adopted for metafile data. This is to impose no requirements on the design data files but only on the metadata records used within design data management systems.

4.7.3 Recommendations - Design Data Management

To address the requirements for a standard interface between design tools and the data management subsystem, it is recommended that an ITC Message be defined. This would support the ability to call for the invocation of these services independent of the chosen management system and to support replacement of the design data management system without necessary change to the design environment or tools. EDA tools could optionally exploit these services.

It is also recommended that the Industry Council commission an organization to work with industry to develop the metadata standard.

4.7.4 Roadmap - Design Data Management

Below are the key roadmap items for design data management.

4.7.4.1 Standard Interface between Design Tools and the Data Management Subsystem

It is recommended that the Industry Council issue a “call to action” to companies and industry groups involved with design data management to establish the semantics and definition of the fundamental design data management services. It should be possible for this to be quickly developed using as the base those definitions currently in use by commercial offerings and within the industry standard groups (RASSP, NIIP, STEP, and WfMC). When complete, this standard set of function definitions should be offered to a formal standards organization (such as IEEE) for life-cycle support and dissemination across the industry.

CFI, or a similar industry consortium, should be commissioned to develop through its membership, the standard ITC Message set for these base services and provide a mechanism for life-cycle support. This could be accomplished either through its own organization, the formal standards body accepting the definitions, or the X-Open consortium. NIIP and its affiliation with OMG should be commissioned to develop the object oriented class members equivalent to the ITC messages with OMG as the preferred life-cycle support organization.

This item is high priority (i.e. in the short term).

4.7.4.2 Design Data Management Metafile Interchange Standard

In order to support both the ability to incorporate multiple design data management systems within an enterprise and the ability to substitute different design data management subsystem(s), a standard must be developed and adopted for metafile data to meet the above requirements.

It is recommended that the Industry Council commission an organization to work with industry to develop the metadata standard.

This item is medium priority (i.e. in the short term).

4.8 Design Process Metrics Management

Design process metrics management encompasses the collection, analysis, and use of metrics about a previous design process in order to predict the needs of and improve on a current design process.

4.8.1 Current Environment - Design Process Metrics Management

Current methods for design process metrics management are necessarily ad hoc with little or no support from the EDA industry. Design groups are finding it increasingly necessary to gather and analyze process metrics about CPU usage requirements, design data storage requirements, memory requirements, and the like in an attempt to predict their capital requirements for the semiconductor technology explosion. Lack of any standard methods or other business decisions result in little or no customer support in this area from the EDA suppliers.

Inconsistent use of quantification terms such as simulation “events” per second, bytes of stor-

age per “block,” etc. make it very difficult to characterize design tool requirements relative to design densities. Inability of tools to report in a consistent manner information about CPU and memory utilization for example, make it a customer burden to develop ad hoc facilities. Lack of standards for metric collection make it impossible to use this data in an automated manner (such as within a design manager) without custom internal development. Lack of rigid metrics for overall design productivity (such as circuits per person month) make it near impossible to analyze opportunities for improvement based on the history of other design projects. Though this is not perceived to be a show-stopper, it is a common complaint voiced by design teams who feel that design productivity will need to improve greatly in order to realize predicted future design densities in products.

4.8.2 Requirements - Design Process Metrics Management

4.8.2.1 Standard for Design Process Metrics

It is highly desired that the EDA industry define a standard definition for a set of meaningful design process metrics that spans all design disciplines and design phases.

4.8.2.2 Standard Interface to Metrics Collection

It is desired that a standard interface to metrics collection be defined and supported by all commercial EDA tools so that standard process related information can be gathered in a standard way for analysis by design teams and/or used within design management systems.

4.8.3 Recommendations - Design Process Metrics Management

In the area of standards for design process metrics, it is recommended that the current work in the industry be combined under an Industry Council-endorsed group and that standard terms and definitions be created for use in metrics collection and reporting systems.

The recommendation to support standard metrics collection is to develop a standard set of ITC messages. This will enable the collection of various metrics based on capturing such design activity as tools being used, by which users, for what amount of time, etc.

4.8.4 Roadmap - Design Process Metrics Management

4.8.4.1 Standards for Design Process Metrics

It is recommended that the Industry Council endorse and support the activities currently underway at NCSU (partly funded by SEMATECH) to develop EDA metrics. This work should be expedited with planned deliverables that can be adopted within the industry. In addition, it is recommended that the work at MIT under the RASSP program in this area also be evaluated and the results used appropriately.

This item is medium priority (i.e. in the short term).

4.8.4.2 Standard Interface to Metrics Collection

It is recommended that the Industry Council commission CFI to develop a standard set of ITC messages and an approach for commercial EDA tools to report these metrics in a standard format.

This item is low priority (i.e. in the short term).

4.9 Design Tool Management

Electronic design tools and systems have evolved rapidly over the last decade (though not as fast as product technology itself--hence the “technology gap”). Change will be a *constant* part of life for designers, the EDA industry, and the CAD system integrators. New requirements such as those in the National Technology Roadmap for Semiconductors (NTRS) and IPC OEM Requirements (IPC) will drive the development of new tools that are not even envisioned today. New design tool technology must be inserted into the design process as soon as it is available in order to meet leading edge technology demands. This changing environment places a challenge on developers to provide tools that are easily installed and encapsulated into the environment(s) chosen by the designers and CAD integrators.

4.9.1 Current Environment - Design Tool Management

Using EDA design tools today is problematic in that each tool has its own, often unique, requirements for proper execution. Having to remember the installed tool location, executable name, command syntax, and required and optional arguments necessary to invoke a tool often leads to errors and decreased productivity. In addition, a given tool may require execution from several different environments (e.g., launch tool from the graphical desktop or from a workflow manager). Today, encapsulation and integration of tools is unique for each tool in every environment. Development and maintenance of such personalized tool integrations is costly and error-prone, and can lead to longer time-to-market for the electronic product developers.

Therefore, there are several aspects of tool management that must be considered. The first is the management of individual EDA tools, that historically have been limited to providing a set of services to the individual design engineer. These services include the launching and stopping of tools, encapsulating the required inputs and outputs for a given tool, and grouping tools into logical clusters of tools (e.g., those tools used in design verification). Many of the EDA frameworks available today provide these capabilities. Also, managing an EDA tool has often been viewed from the perspective of a single user. However, in today’s environments, tool management must be viewed from a more global perspective. Larger design teams are working with a greater number of tools, spread over a larger geographical area. As the complexity of designs grows, larger design data sets will exist across a larger number of machines. Tool managers must also be able to manage and track the status of tools on remote hosts and in heterogeneous environments.

The second aspect of tool management is associated with the management of computer systems. The rise in complexity of the design problem has led to an increase in the number of design tools used by most users. In large design environments, installing and managing these tools has become a problem large enough to impact designer productivity. The methods for installing, licensing, and upgrading tools differs with each vendor and is often inconsistent even between tools from a single vendor.

4.9.2 Requirements - Design Tool Management

This section summarizes the requirements for inter-tool communication, tool encapsulation, and system management guidelines.

4.9.2.1 Standards for Inter-Tool Communication (ITC)

To adequately manage tools, a Tool Manager must have the ability to start, stop and determine the current status of individual tools. One way to accomplish this would be to build a monolithic environment containing all the tools the manager wishes to control. However, this scenario does not provide the flexibility needed for today's multi-vendor environments. A more efficient alternative is to provide a means of communicating between tools, via dynamic inter-tool communications. Since users need to manage tools from a variety of vendors, a standard message passing mechanism together with a standard message dictionary are required.

4.9.2.2 Standards for Encapsulating Tools

A tool encapsulation standard is required to capture the structural aspects of a tool. These include the tool's executable name and location information, tool class, operating environment requirements, and input and output requirements (along with their data types). It must be possible for this standard to enable highly-automated encapsulation of tools into multiple usage environments including

- Graphical user interface environment (e.g., CDE)
- Workflow management environment (e.g, a workflow manager engine)
- Vendor-specific environment (e.g., a framework)
- Non-graphical environment (e.g., tool execution via script files or command-line invocation).

For example, in a graphical environment, the encapsulation of the tool may result in the user being presented with a dialog box and asked to enter information such as the location of a design object. To enable encapsulating the structural aspects of tools (e.g., inputs, outputs, launch mechanism), a common tool encapsulation method is required. This standard must support tools that operate in either graphical or non-graphical environments.

4.9.2.3 Standards For License Management

Because of the number of tools (and potentially multiple versions of each) required from different companies and the sheer volume of licenses to be managed across multiple designers and computers, there is a requirement for standard(s) for license management.

Ideally, it is desired that all EDA tools use a single license manager and that all use it in a consistent way so as not to require the customer to create and maintain multiple key files and processes.

4.9.2.4 EDA Systems Management Guidelines

Another aspect of tool management is from the system management perspective. The rise in complexity of the design problem has led to a proliferation of EDA tools that are often aimed at specific problems. In many user environments, the number of tools (and multiple versions of those tools) has increased dramatically, and this is expected to continue. As a result, managing the installation and licensing of these tools has become a problem large enough to impact designer productivity.

Users need advice on the best methods for managing their EDA systems. While a standard may not be appropriate, a set of guidelines and best practices should be collected and shared with users. This document could also identify additional opportunities for standardization. For example, the method for installing tools differs with each vendor, and is often inconsistent even between tools from a single vendor. This document could recommend a common tool installation procedure be developed, such as the one used by tools in the Windows environment.

4.9.3 Recommendations - Design Tool Management

A number of vendors have implemented messaging facilities, that for the most part are used within a single vendor's tool set. The ToolTalk messaging facility, included as part of CDE, is vendor independent and has been endorsed by CFI as a standard inter-tool messaging mechanism. An additional advantage of ToolTalk is that the software will be provided by hardware vendors as part of their OS offerings. This removes the burden on tool vendors to implement another messaging facility. This standard must be made available on both UNIX and the Windows-PC environments.

However, using a common messaging facility by itself is not enough. Equally important, is a common set of messages that tools can use to communicate. CFI has developed a draft EDA Message Dictionary standard that should be used as a starting point.

With regard to standards related to tools integration, the CFI Tool Encapsulation Standard (TES) is the only existing standard in this area. TES has been evolving to meet EDA industry needs, and has recently been extended to better support automatic encapsulation of tools into the CDE environment and to meet requirements such as those identified above. It should become standard operating procedure for EDA vendors to ship TES files with their products. All other tool creators and CAD integrators should also use TES to aid in integration of tools into the design process.

In the area of license management, it is recommended that the Industry Council commission the creation of an RFT for a commercially available license manager that meets the EDA community functional and business needs. Since the industry has more or less already converged on a single license manager product, it is anticipated that the functional capabilities of that product be adopted as a base standard, and modified only where there is a specific and justified EDA industry need (e.g., cost).

In addition, a consistent policy for its use must be developed. Further, it is recommended that upon selection of the winning RFT bidder that the Industry Council do whatever it can to enforce the industry-wide use of this product.

It is recommended that the Industry Council commission a team of EDA companies to develop a standard use model for the winning license manager product so as to provide a consistent license management process for EDA customers independent of their tool choices. Life cycle support for this standard use model could be commissioned to EDAC or another industry organization of choice.

For the system management aspects of tool management, we recommend a set of guidelines or best practices be collected. The guidelines should cover items such as the tool installation, file location strategies, license setup, and managing multiple released versions of a tool. Guidelines should be available for both the UNIX and Windows-PC environments.

4.9.4 Roadmap - Design Tool Management

This section outlines the recommendations for inter-tool communications, tool encapsulation, and systems management guidelines.

4.9.4.1 Intertool Communications: Adopt ToolTalk, Provide Windows Interoperability

The recommendation is to adopt ToolTalk as the standard for communications between tool managers and EDA tools. ToolTalk together with a standard EDA Message Dictionary can meet the requirements for inter-tool communications. The ToolTalk messaging facility should be supplied by UNIX hardware vendors as part of their OS offering. It must also be made available on the Windows-PC platform. Since the Message Dictionary is EDA specific and its contents will change over time, it should be developed and maintained by an EDA organization such as CFI. Also refer to 4.2 "Design Tool Communication" for additional discussion on this topic.

This item should be adopted as high priority immediately (i.e., in the short term).

4.9.4.2 Tool Encapsulation: Adopt CFI TES

Since CFI TES meets all the documented requirements for support of tool encapsulation into the various EDA desktop, workflow management, and vendor-specific usage environments, it should be immediately adopted in the UNIX environment. Standard TES files should be shipped by EDA vendors at their next major release of their tools to enable the automatic installation of new tools into the above EDA environments, and to minimize impact on CAD integrators when new tools are installed for use.

This item should be adopted as medium priority immediately (i.e., in the short term) for the UNIX environment.

While TES files may not be directly useful to aid installation of tools in the Windows environment, they can be very useful in integrating UNIX and Windows tools into design (workflow) managed environments which span multiple operating environments. Creation of TES files for Windows tools is lower priority than for UNIX tools.

4.9.4.3 License Management: Establish Base EDA Industry License Manager/Use Policy

While there are some reasonably effective license management technologies available today to support the management of design tool software, the policies for their use are not standardized and cause much consternation in the user community. This task is to establish a common EDA industry-wide tool license manager, and a common policy for its use. This license management standard is focused on the UNIX environment in the short term, but eventually becomes important in the Windows environment as well. Extending common license management standards such as floating licenses into all operating environments is highly desirable.

This item should be adopted as high priority in the short term.

4.9.4.4 EDA Systems Management Guidelines

The recommendation for this item is to collect a set of guidelines or best practices to advise users with regards to EDA systems and tool management. This document should be created by users working closely with the tool vendors, and owned by a user group such as USE/DA.

This item should be adopted as low priority immediately (i.e., in the short term).

4.10 Resource Management

With the advance in technologies, the design complexity is increasing at a compound rate of over 50% per year. SEMATECH predicts that the design team size might approach 270 in 1998 and 600 in 2001. These large design teams will require an environment of hundreds of networked desktops and servers, a large number of Electronic Design Automation (EDA) and productivity tools and, millions of data files residing on tera-bytes of disks.

In the future, the EDA users will face several key problems, including:

1. How to access and manage these resources (machines, disks, licenses, data...).
2. How to efficiently utilize these resources.

“The challenge is managing the exploding complexity in the design environment”

4.10.1 Current Environment - Resource Management

The current design environment consists of heterogeneous hardware platforms and software packages. The different hardware and software generally do not communicate with each other and usually an internal CAD group has the charter to make all of these components work together. Although the hardware is networked together, access to a given machine can be very different from accessing another because of the operating system, directory structure, license server, and so on. To overcome these difficulties, the CAD group develops “glue” software to link the different software packages to form automated design flows to be used by the designers. Further automation is customized by individual users through the creation of “ad hoc” scripts.

The design environment resulting from these “band-aid” solutions can be very fragmented and highly individualized. Automation can be achieved at the individual level, but there exists no mechanism to share data or tasks at the workgroup level.

Standards will be greatly beneficial in prescribing uniform ways and best practices or guidelines to organize and access the resources in a design environment.

4.10.2 Requirements - Resource Management

A study on the ratio of computers to users over time reveals, that in the future, each user will have at his/her disposition a large amount of computer resources: the challenge is how to harness these resources. In the 1970's, a group of 50-100 people shared a mainframe for their computing needs with a ratio of perhaps 1/50-100. In the 1980's and with the introduction with the workstation, most of the users have a dedicated computer and the ratio became 1/1.5. In the 1990's, the hardware cost has significantly dropped and the design complexity has increased dramatically resulting in a ratio of 2-5 computers for each user. This trend is expected to continue in the year 2000 with 10-15 computers available to each user.

The requirement is to transform the hundreds of networked computers, and the corresponding data and tools, into a single and "virtual" resource pool for the users.

These resources include computers, disks, networks, tools, and data. They need to be tightly coupled with each other in order for the users to achieve the full benefit.

4.10.2.1 Support for Load Balancer/Job Scheduler

A Load Balancer/Job Scheduler is required to assist the designer in managing the queueing of the jobs and in balancing the load of the computers both for interactive and batch jobs. This facility must allow the user to efficiently utilize all available computer resources. The user should be able to dynamically reconfigure priorities based on job types, user profile, project, time, and machine configuration.

4.10.2.2 Support for Uniform Way to Organize the Design and Related Data

A uniform way to organize design data and its hierarchy is need which can present users throughout the enterprise with a uniform view of the design data independent of which computer they log on. This feature is important since users and the jobs they create, must be able to access any node in the network, and a consistent view of the data is extremely important.

4.10.2.3 Support for Workgroup Design Data Management

The design data is the most important resource in the design environment. There must be a facility to manage design data through the design cycle, and its evolution through the workgroup.

A release and archival facility for the design data is needed.

4.10.2.4 Support for "Design Warehouse"

In the complex circuit development environment, the user is presented with a large amount of information to be analyzed, in order to make decisions and to take action. A "design warehouse" for the design data, is required to facilitate this task. The "design warehouse" would collect, sort, organize, and present data in views appropriate for the different user classes, and with appropriate security measures. Based on predefined rules, the users will be presented with options on how to proceed.

4.10.3 Recommendations - Resource Management

There are no specific standards activities recommended at this time. However, the above areas should continue to be monitored regularly as technology evolves.

4.10.4 Roadmap - Resource Management

There is no specific roadmap for design resource management in this version of the roadmap.

This page was intentionally left blank.

5 The Design Information (Design Data Representation)

This chapter includes descriptions of important design information (data) generated and/or used in each of several key activities related to the electronic design of components and systems as described below.

The design information environments for designing cores and chips, MCMs, and boards are similar in many ways, as are the requirements for future support. Environmental comments, unique to a type of packaging, are identified and discussed within the relevant design category. Therefore, after some debate, it was decided that for the purposes of discussing the relevant EDA standards, it would be appropriate to discuss the major design phases in two categories called “system design” and “detailed design”. This is not meant to imply a particular design process or methodology, but rather to enable groupings of related standards. These environments and some rationale for this grouping are described in some detail below.

In order to eliminate redundancy, requirements that were common across each of the above categories were grouped together in the section “common topics”. At the end of this chapter, the technology rules and models required to support all phases of design are discussed in the section on “design technology re-use”.

Therefore, this chapter is organized as follows:

- 5.1 "Common Topics Across Design Information"
- 5.2 "Common Topics Across Design Steps"
- 5.3 "System Level Design"
- 5.4 "Detailed Design"
- 5.5 "Design and Technology Re-use"

5.1 Common Topics Across Design Information

The topics discussed in this introductory section are important independent of the design step involved, and also independent of the design information represented. The topics represent additional requirements to the unique requirements stated in the following sections (5.3.4 "Roadmap - System Level Design" and 5.4.4 "Roadmap - Detailed Design"). They are stated here in order to be more concise, and avoid repeating this information in both sections.

5.1.1 Incremental Processing

This section addresses incremental processing: the current environment, requirements, recommendations, and roadmap.

5.1.1.1 Current Environment - Incremental Processing

The rapid growth of design size as indicated in the NTRS Roadmap implies a corresponding growth in the amount of design data. During the execution of the design process, design engineers frequently make small changes in the design (small relative to the entire design), and then the effect of those changes is evaluated.

What appears to be a “small design change” by the engineer actually affects a “large file” (in the case of a large SDF file or EDIF file, for example). The amount of time needed to reprocess these large files is becoming prohibitive, and is not in proportion to the designer’s changes.

5.1.1.2 Requirements - Incremental Processing

This section discusses the known requirements in the area of incremental processing.

5.1.1.2.1 Support Incremental Processing in the Design Representation Standard

The above situation drives a need for standards that support the relatively high performance requirements for interactive programming interface communication between EDA design tools. For example, the interaction between logical and physical design tools when performing timing verification on a design. This requirement is related to 4.2.4 "Roadmap - Design Tool Communication", but has EDA domain-specific connotations (e.g., message passing of items such as “highlight net”).

There is *also* a need for incremental file-based communication for logical connectivity and physical design data. This type of incremental file-based approach is needed to support the transfer of design data between environments where file sharing or inter-tool communication is not possible, feasible, or practical. An example of where this is a requirement is the interface between an OEM customer and an ASIC design shop where direct data sharing is often not possible. The information needs to be transferred, and exchanging messages between tools may not be practical. In such cases, file-based transfer may be a preferred method of moving the design data to where it has to go. For additional justification of the need for incremental processing of design data, refer to 5.4 "Detailed Design", and specifically Table 5.1: "Impact of Design Size on Design Processing Times".

There is a clear requirement for design representation standards such as 5.4.2.1 "Standard Detailed Design Representation" to support incremental processing of design changes (or engineering changes). Incremental processing as used here applies to the entire design cycle and any phase within the design cycle. This requirement needs to be implemented via appropriate programming interface and file-based solutions to meet the total set of requirements.

Support for incremental processing must be architected into the total design process.

5.1.1.3 Recommendations - Incremental Processing

It is recommended that support for incremental processing be designed into *all* of the proposed PI and file-based design representation standards. Refer to 5.4.2.1 "Standard Detailed Design Representation".

5.1.1.4 Roadmap - Incremental Processing

5.1.1.4.1 Support Incremental Processing in the Design Representation Standard

Incremental processing must be supported in 5.4.4.1 "Converged Industry Standard for Logical Connectivity" and for each of the follow-on design data representation standards. The base standard and any extensions should be monitored so that the strategic standard can include the necessary information. EDA vendors should support incremental processing of the strategic design representation standard as soon as it meets basic requirements.

This item should be adopted as high priority immediately (i.e., in the short term).

5.1.2 Hierarchical Processing

This section addresses hierarchical processing: the current environment, requirements, recommendations, and roadmap.

5.1.2.1 Current Environment - Hierarchical Processing

As was stated in 5.1.1.1 "Current Environment - Incremental Processing", the rapid growth of design size as indicated in the NTRS Roadmap implies a significant growth in the number of designers on a design team. In order to manage very large designs, it is a common practice to break up the design into pieces so that the pieces can be designed independently and yet be put together in a system design for system level and detailed processing. There is therefore a corresponding growth in the number of design objects being designed by the design team. This hierarchical design approach of "divide and conquer" must be supported, particularly for large complex chip designs.

5.1.2.2 Requirements - Hierarchical Processing

This section discusses the known requirements in the area of hierarchical processing.

5.1.2.2.1 Support Hierarchical Processing in the Design Representation Standard

There is a requirement for design representation standards to support hierarchical processing of design changes (or engineering changes). Hierarchical processing as used here applies to the entire design cycle and any phase within the design cycle. This requirement needs to be implemented via appropriate programming interface and file-based solutions to meet the total set of requirements.

There is a requirement for EDA tools, independent of the design phase, to support hierarchical design representation and processing of design elements. This requirement has some similarities to requirement 5.1.1 "Incremental Processing", but is focused on handling design information across the design hierarchy.

Support for hierarchical processing must be architected into the total design process.

5.1.2.3 Recommendations - Hierarchical Processing

It is recommended that support for hierarchical processing be included in *all* of the proposed design representation standards. Refer to 5.4.2.1 "Standard Detailed Design Representation".

5.1.2.4 Roadmap - Hierarchical Processing

5.1.2.4.1 Support Hierarchical Processing in the Design Representation Standard

Hierarchical processing must be supported in 5.4.4.1 "Converged Industry Standard for Logical Connectivity" and in each of the follow-on design data representation standards. The base standard and any extensions should be monitored so that the strategic standard can include the necessary information, and EDA vendors should support hierarchical processing of the strategic design representation standard as soon as it meets basic requirements.

This item should be adopted as high priority immediately (i.e., in the short term).

5.1.3 Design Object Naming

This section addresses design object naming: the current environment, requirements, recommendations, and roadmap.

5.1.3.1 Current Environment - Design Object Naming

There are many useful design tools available that use naming conventions that have been developed to support legacy naming conventions for EDA design objects (such as net name, block name, etc.). In a multi-vendor design system environment, this frequently causes significant confusion, particularly when several tools that use different naming conventions are in a tight design loop.

Work in the area of name mapping is required to allow tools from different EDA vendors to map between the names they use for the same EDA objects. This capability is a prerequisite for doing many operations (such as cross-probing), to support design reuse, and for effective data transfer between tools.

Most of the design objects that EDA tools want to access have a "name" attribute. However, in a given vendor's design tool or system, this name may have limitations (e.g., on the length or character set that can be used for a name). Further, the specific rules for naming objects can vary from application to application, and from vendor to vendor. This creates a very complex mapping problem in today's multi-vendor environments and confirms again the difficulty of writing translators to convert files from one form to another. This problem is exacerbated when the information passed from one tool to another is in a key design loop where inter-tool communication is desirable.

5.1.3.2 Requirements - Design Object Naming

A standard for design object naming is desired in the implementation of the roadmap for the design data representation (see 5.4.2.1 "Standard Detailed Design Representation"). This standard should define the strategic design object naming convention, and support for legacy applications, so there are defined mappings to/from that standard naming convention to vendor-specific naming conventions. This naming standard must also meet the needs for communicating mappings of named objects via standard inter-tool communication mechanisms (see 4.2.2.1 "Standards for Inter-Tool Communications").

5.1.3.3 Recommendations - Design Object Naming

It is recommended that a new standard naming convention be developed for EDA design objects. It is also recommended that in order to support coexistence and migration to that new standard, a set of mappings from today's naming conventions be documented so that the specification can define how existing vendor applications naming conventions map to that standard.

5.1.3.4 Roadmap - Design Object Naming

5.1.3.4.1 Standard Design Object Naming Convention

Develop a standard design object naming convention for the next decade, and define how existing vendor applications naming conventions, map to that standard. Ensure that 4.2.2 "Requirements - Design Tool Communication" and 5.4.2.1 "Standard Detailed Design Representation" will both be included in this work. Refer to those sections for further information.

This item should be adopted as high priority immediately (i.e., in the near term).

5.2 Common Topics Across Design Steps

The topics discussed in this introductory section are important independent of the design step involved, and are global requirements that must be met in addition to the unique requirements stated in the sections that follow (5.3.4 "Roadmap - System Level Design" and 5.4.4 "Roadmap - Detailed Design"). They are stated here in order to be more concise and to not repeat this information in other sections.

5.2.1 Timing Information

This section addresses timing information: the current environment, requirements, recommendations, and roadmap.

5.2.1.1 Current Environment - Timing Information

Many current timing driven design tools interface to the de facto Standard Delay File (SDF) batch timing information file. The tools that access the SDF read the entire file and correlate the timing data with separately specified connectivity or structural data describing the design. The existing standards (e.g., EDIF, CFI DR, AP210) do not yet (formally) include the timing data.

This approach will not be adequate for the large designs of the next decade or even for the more immediate term. These very large chips will require many designers on complex chip designs and system designs, and those designers may well be spread across geographic regions. The delay due to interconnect will be a very significant portion of the delay in .35um technology. Complex large high-performance chip and system design, often requires the inclusion of timing design as an embedded part of the early high level design. Timing has become a key constraint in constraint-driven design.

Integrated Design Environment

In an integrated design environment, where structural design changes or interconnect changes can be made, the engineer will need the delay calculation and timing analysis to be done on only the changed or effected area of the design (or design object). The engineer will want to see the results back annotated on the source form. This will require that the timing data be available in the procedural interface form on a hierarchical and incremental basis, and across the geographically dispersed computer network.

Separated Environment

In a separated environment, such as an OEM designer interfacing with an ASIC foundry doing the back end physical design, the OEM designer will need the capability to interface floor-planning timing constraints, and timing estimates, with the structural data. This data is the basis for the designer's initial verification. As hierarchical and incremental processing is added across this interface, the OEM designer will need the capability to transfer incremental timing changes that correspond to the incremental structural changes in an integrated fashion. After physical design is completed, the final timing data must be sent back to the original OEM designer, without requiring that the 10's of megabytes of structural data be included.

5.2.1.2 Requirements - Timing Information

5.2.1.2.1 Timing Information in Detailed Design Representation

There is a requirement for design representation standards such as 5.4.2.1 "Standard Detailed Design Representation" to support EDA tools so that they can share and/or exchange timing information, in a standard inter-operable way across all design phases.

Each EDA timing driven design tool, should be driven from, and constrained by, the timing performance and implementation requirements of a design. There needs to be a consistent architecture for the following:

- timing design information entry
- timing constraint capture
- timing analyzer use
- high level early interconnect estimation
- manipulation of cell hierarchy and timing libraries
- timing information storage
- timing design management and change control
- other design activities using timing driven design information.

During the early phases of the design, timing may be a matter of re-use information, area based estimates, or other design requirements. Later, after the physical design, timing is fully determined by the structural, behavioral, and physical detail of the design. The timing driven design process requirement, therefore, is embedding the timing design into the usual mixture of Top Down, Bottom Up, and Middle Out design process paradigms.

- Top Down - Timing needs to be imbedded in the design of the overall target system functions and performance targets.
- Middle Out - Timing needs to be propagated upward to higher levels of design and downward to more detailed level of design.
- Bottom Up - Timing needs to be imbedded in the early design of the underlying chip technology, low level design functions, and cell and transistor libraries used to implement the low level design functions. This requirement includes interfaces to technology design for joint development of accurate models. These lower level timing parameters need to be abstracted, and propagated up the hierarchy.

Once the initial design steps have been completed, timing design must continue to be embedded in the incremental design of the hierarchical chip.

These requirements include support for timing representation so that static timing analysis and other calculations can be performed hierarchically on a design element, without reprocessing an entire design when possible to evaluate a change in only one element in the hierarchy. Some technologies have unique timing information representation requirements, such as the rise time dependent delay requirements of CMOS.

The following paragraphs identify information and supporting infrastructure that is required in properly support timing design:

- Timing Characteristic Type

A timing characteristic type identifies the role of the timing information in the design. The specific terms included in this class should, include timing characteristics (such as given in the DIE-Timing latest draft document) including delay, setup, hold, pulse width, cycle time, period, rise and fall time).

- Application-Specific Timing Information

Application-specific timing information identifies the designer's intended purpose, or origin of a set of timing values. An important set of application-specific information to include are those that identify timing design management information, (e.g., required, budgeted, analyzed, actual, measured, and current). Each of these can be presented in a statistical manner with nominal, mean, min-max, sigma, confidence level, skewness, and kurtosis.

- Variational Timing Information

Variational timing information provides a means for defining timing information values as a function of other design requirements, constraints, and parameters. The expressive capabilities should encompass those of the Delay Calculation Language (DCL). The expressive capabilities should include functionality required to support physical based delay estimate calculations, including high level abstract floorplanning. Capabilities should also include a facility for defining the sensitivity of timing information to design process parameters and value set changes. For example, an ability to determine the sensitivity (rate of change) of delay to design and process parameters.

The above items will provide the capability to calculate delay using varying electrical parameters. These electrical parameters include source and load information that may be associated with circuit transistor, switch, gate and block levels of design.

- **Interconnect Net and Path Identification**

Means should be provided to identify the timing design related function, status, and criticality of the elements of net and path interconnect within a hierarchical chip design. Identification should include facilities such as nets, net segments, paths, ports, and signals. The specific function of the interconnect should be identified, (e.g., clock, bus, power, and ground). The criticality of the interconnect should be identified as critical or noncritical.

5.2.1.2.2 Delay Calculator Language (DCL)

The requirements in support of the proposed standard Delay Calculator Language (DCL) are as shown in the Standard Delay System Objectives Version 1.1, 11/8/94:

- Allow silicon foundries to describe delay equations for ASIC libraries, and in a single way which can be used by any set of compliant CAD applications and vendors
- Allow CAD vendors to interface to delay calculation for specified design elements using a single interface to the foundry supplied equations
- Account for interconnect capacitance as well as gate capacitance, which implies more complex equations; more CAD tools are now required to access delay calculation with these complex equations
- Provide a Delay Language compiler which creates a compiled form of the delay expression language, which when executed with a specific net description, can calculate all necessary delay characteristics of the net and associated gates. This then provides the *same* set of delay values to each vendor tool. At the same time, the compiled DCL hides proprietary descriptions of process-related performance which ASIC vendors often wish to keep confidential.
- Provide a programming interface (PI) to a compiled form of delay equations described in the delay equation expression language
- Delay equations distributed from the semiconductor vendor must be protected from accidental or intentional loss of intellectual property rights
- The calculation from the delay equations must be of high enough performance to support both the batch calculation of all delays, and the incremental calculation of individual nets within design applications such as synthesis. This calculation is to include both pre-layout and post-layout phases of the design process

In addition to the above, DCL is required to support arrays and program loops (i.e., FOR loops) to support transmission line analysis, and it needs to support the scope of DIE-T and IBIS standards to meet system delay and timing driver design requirements.

5.2.1.2.3 IBIS and DCL Integration

DCL is a specification and PI for timing information in support of delay calculation and other timing analysis tools for chip design as described in the previous section. IBIS (I/O Buffer Information Specification) is an emerging standard for electronic behavioral specification of

integrated circuit input/output analog characteristics in support of higher level package analysis (above the chip level). IBIS specifies a consistent software-parsable format for essential timing information. With IBIS, simulation tool vendors can accurately model compatible buffers.

An approach to standards in this area is required which integrates the DCL and IBIS capabilities into a single set of standards. This is required to support delay calculation and signal integrity at the chip level and above.

5.2.1.2.4 Common Delay Calculation

A common delay calculation capability is required to provide consistent timing information across applications, at the chip level and above.

5.2.1.3 Recommendations - Timing Information

In the area of timing information in detailed design representation, it is important that the processing and design time to achieve the necessary information transfer and timing correlation be minimized. This is due to:

- The extreme significance of the interconnect on the delay characteristics of the design
- The pervasiveness of the timing impact throughout the design process (i.e., in both system level design and in detailed design)
- The increasing number of large systems that will be designed across geographic dispersed regions.

Therefore, the current de facto standards like SDF need to be integrated into the proposed standard design representation (both procedural and file-based interfaces). The transfer of timing data between design activities should be supported, integrated with the structural data or independent of the structural data, to support the many different methodologies and design scenarios.

To meet the requirements in this area, it is recommended that

- Timing information be added to the detailed design representation standards described in 5.4.4 "Roadmap - Detailed Design"
- The DCL and IBIS standards must be integrated to meet the above requirements
- The CFI/OVI Delay Calculation Language and PI effort be completed, and then extend this capability beyond ASIC packages.
- Appropriate interfaces to TCAD be developed (not addressed in this version of the roadmap).

5.2.1.4 Roadmap - Timing Information

5.2.1.4.1 Integrate Timing Information into Design Representation Standard

Timing Information currently supported by standards like SDF should strategically be supported by 5.4.2.1 "Standard Detailed Design Representation". Extensions to SDF should be monitored so that the strategic standard can include the necessary information, and EDA vendors should migrate to the strategic Design Representation Standard as soon as it meets requirements for timing information.

This item should be adopted as high priority immediately (i.e., in the short term).

5.2.1.4.2 Integrate DCL and IBIS

DCL and IBIS capabilities should be integrated into a single set of standards. This is required to support delay calculation and signal integrity at all levels of package.

This item should be adopted as high priority immediately (i.e., in the short term).

5.2.1.4.3 Complete the CFI/OVI Delay Project and Extend Beyond ASIC Packages

The delay project to demonstrate the applicability of DCL and the PI to delay calculation must be completed.

This item is high priority in the short term for ASICs, and medium priority for PCB and MCM packages.

5.2.2 Simulation and Test Control

This section addresses standards for simulation and test control: the current environment, requirements, recommendations, and roadmap.

5.2.2.1 Current Environment - Simulation and Test Control

A design team uses simulation to verify logic correctness, interfaces between entities, verification of synthesis translation, and to determine fault coverage. The common Simulation and Control standard should address all of these activities such that a design team can reuse test-benches and test vectors between simulators throughout the entire design cycle.

There are a number of different simulators that can be used during the system level and detailed design phases. For most of these simulators, there is a unique way to specify the stimulus and expected response, control the simulation, control the debug of the design, and collect the results from the simulation. Moving this type of information from one simulator to another can be a time-consuming and difficult conversion effort. The lack of a common simulator control language is an inhibitor to simulator "plug and play".

The standard should also provide mechanisms which allow the designer to capture the *intent* of a test bench or test vector segment in a fashion similar to how we document the design itself.

In addition to the lack of a simulator control standard, there are also several test vector formats being used to drive testers in the manufacturing test environment. Moving test information such as stimulus and expected response, and other control information from one test environment to another is also difficult.

5.2.2.2 Requirements - Simulation and Test Control

Standards are needed in this area that addresses the requirements of both simulation and manufacturing test.

5.2.2.2.1 Standard Simulation Control Specification

A standard for specifying stimulus/response and other simulation control information is required. This standard must encompass support for the stimulus and expected response, control the simulation, control the debug of the design, and collect the results from the simulation. It must address analog, digital, mixed analog/digital and mixed language designs.

5.2.2.2.2 Standard for Test Control Specification

A standard for specifying stimulus/response and other manufacturing test control information is also required which accounts for the capabilities of commercial testers. This standard must encompass support for the stimulus and expected response, control the test, control the execution of the test, and collect the results from the test. It must address analog, digital, mixed analog/digital and mixed language designs.

5.2.2.3 Recommendations - Simulation and Test Control

A new industry standard should be developed to support a common simulator control specification. This should be a single standard that addresses the simulator environment during system and detailed design, as well as the manufacturing functional test requirements.

5.2.2.4 Roadmap - Simulation and Test Control

5.2.2.4.1 Converged Industry Standard for Simulation Control Specification

A new industry standard should be developed to address the requirements for a common simulator control language to meet the requirements stated above.

The IEEE DASC Simulation Control Language group intends to start paperwork for formal submission (completion date before 06/96) on this Standard Simulation Language. The work of this group should be considered in the development of the industry standard.

This item should be adopted as medium priority immediately (i.e., in the short term).

5.2.2.4.2 Converged Industry Standard for Test Control Specification

The simulation control standard developed in the above step, should address not only simulator control requirements, but enable the subset of simulator control information (e.g., stimulus and expected responses, and loop controls, etc.) to be included as part of the key interface to manufacturing. This would allow testing of the manufactured design object in the same way that the design was simulated during the design phase.

This item should be adopted as medium priority immediately (i.e., in the near term).

5.3 System Level Design

This section addresses system level design: the current environment, requirements, recommendations, and roadmap.

5.3.1 Current Environment - System Level Design

The system design step covers the architectural and high level design phases of electronics system design, and creates the high level design plan to be implemented in the detailed design phase that follows in the next section.

The architectural design phase is typically characterized by very high level design activities, leading towards the establishment of overall partitioning into physical packages, and technology selection for each of the major design objects of the product architecture. Design tasks include evaluating alternative architectures, assessing performance or throughput of candidate architectures, performing various hardware/software trade-off analyses, considering re-use of previously designed objects as part of the architecture, starting of hardware/software co-design, and considering life-cycle costs of various early design decisions. Architecture design and the relationship to software co-design is discussed in 6.4 "Software Design Interface (Hardware/Software Co-design)". Examples of current related work in this area include SpecSyn and the Ptolemy project at the University of California, as well as in the RASSP program.

During the high level design phase, the functional details of each of the major design objects from the architecture phase are modeled functionally (or behaviorally) in system level design specification languages. The languages typically used include: VHDL, Verilog, C, or possibly a proprietary specification language. The entire system is evaluated via simulation or other functional evaluation tools to establish confidence that the architectural design concept meets key design function and timing objectives. As functionality is refined, preparation for the detailed design phases gets underway. High level floorplanning and synthesis can be used for chips to generate an implementation level design for the selected IC technology, and to generate or capture constraints on implementation.

Also during the system level design phase, test strategies such as BIST, traditional stuck fault, chip in place, and delay testing are developed.

5.3.2 Requirements - System Level Design

Support for VHDL and Verilog standard hardware description languages is required, as well as any future HDL representations for architectural and system design. A common interoperable interface from any HDL, is required to support independent EDA design tool selection and use. An example of this is the effort by the OMF to develop an HDL-independent simulator interface. Any simulator that uses this interface can simulate models written in any HDL compliant to the simulator interface.

A key enabling technology for very large designs and designs that re-implement previously developed designs has to do with support for architecture design specification and associated re-use. As the popularity of architecture and high level design using hardware description languages such as VHDL and Verilog continues to increase, the potential for re-use of this work

increases. Standards to support this are still emerging, and design representation technology needs to be established to encompass such information as design specification guidelines for encoding the various simulation models (architecture level, RTL level, implementation level, etc.), as well as functional test vectors, and other design data that was developed in the original version of the design. What is a chip today will only be a part of a chip (i.e., a “core”) tomorrow. What is a large subassembly today, will be a chip tomorrow. While this requirement is listed in the architecture phase, it applies to all of the design phases, in the sense that re-use of previous design information is required.

In addition to standards that support HDL specification of designs to support re-use, there is a challenge to enable existing legacy designs to be available for re-use in cases where there is no HDL specification. Tools and techniques for generating models of implementation level designs are needed to support re-use of such designs.

5.3.2.1 HDL Standards to Support Synthesis

While there may be compelling reasons (size of customer set, etc.) for having more than one design language (i.e., VHDL, Verilog, C), there are also some compelling arguments for having a standard set of synthesizable primitives, or subsets of HDLs, along with standard ways of constraint setting and passing of this information to all applicable tools in the Design Environment. The idea of a standard set of HDL primitives is attractive in that very high level design tools could target the HDL primitives as an output, (i.e., a design point from which design synthesis is effective, and synthesis tools could target the HDL primitives as an input, from which design implementation is effective).

Standardizing on a common RTL subset and other higher level HDL primitive types could provide a foundation for significant future innovations in systems design. This concept is similar to the software analogy of programming in a high level language versus assembler language. We must develop higher levels of synthesizable design objects in order to become more productive at producing designs, and in producing technology implementations of designs. Having higher levels of synthesizable design objects will make it possible in the very front end of the design process to generate the higher level design objects, and in the detailed design phases, those objects can be mapped into various detailed design implementations at the technology level.

5.3.2.2 Standard HDL Interfaces to Design Analysis Tools

There is a need for various design analysis tools, including simulation (and other tools) to become more independent of the HDL chosen for doing design specification. There are two important scenarios with similar, yet distinct requirements:

- The “original design scenario”

In situations where *original* design is being performed, design teams have a requirement to choose their preferred HDL in which to design, simulate, and eventually produce the design object. This is the typical scenario of the design team in a component supplier company, who have various design teams who may use different HDLs for a variety of reasons. They may choose VHDL or Verilog based on familiarity and experience or other business reasons. The same scenario is true in large system design companies where multiple HDLs are used for a variety of reasons. In any case, if the design object is reusable

and becomes a building block for use in higher level design, there are additional requirements; for additional information on function reuse, refer to 5.5.2.3 "Standards for Reusable Functions". When designing a single design object, the design team chooses an HDL, and the simulators which support it, and performs design.

- The “reuse scenario”

In any design situation where teams desire to make use of previously designed and reusable design objects such as those from the “design scenario” above, the reusing design teams have requirements to also choose their preferred HDL in which to design and release this higher level of design object, independent from the HDL that any of the building blocks may have used. That HDL may be the same or different than the HDL used to design some of the reusable building blocks.

There are no specific requirements for VHDL and Verilog interoperability within a single original design scenario as described above; however, when in the reuse scenario, it is clear that the reusing design team must be able to choose their HDL and simulator independent of the choices of the original design team.

Strategically, this area should address HDL independent fault simulation and test analysis of hierarchical, mixed-level system models.

As extensions are made to HDLs (individually or in a concerted collaborative fashion), for example mixed digital/analog capabilities, there is a requirement that those extensions meet the above standard HDL interface requirement.

5.3.2.3 Standard Controls for Constraint Driven Design

There is a requirement for a standard for specifying design constraints on the design entity being developed. This promotes interoperability between tools such as synthesis and the HDL languages. Metrics required include specification of target values for total area, total power, maximum path length, specific path point-to-point timing, EMI, etc. In addition, the standard must include the ability to specify cell and power level, and specific physical location for large entities on the package (e.g., large arrays or microprocessors). This standard should be supported in a standard way so that HDL-independent controls can be established for synthesis and the subsequent detailed design activities.

5.3.2.4 Standards to Support Floorplanning

There is a requirement to share information between the synthesis and floorplanning activities performed during system level design. This data includes the specification of such information as cluster specifications, physical boundary requirements, and other floorplanning constraints. This information must also be shared with the detailed design phase so that early high level floorplanning and synthesis can be effectively linked with detailed floorplanning, placement and wiring interconnect, etc.

5.3.3 Recommendations - System Level Design

For system level design in general, it is recommended that a new effort be started to develop a system design standard, using the standards development process recommended in 2.3.2 "Standards Development Process Recommendations". This work should be started by building a system level design information model using VHDL as a guide. This approach is NOT to endorse VHDL as a strategic system design standard, but rather to establish the base information model from the "information content" of the VHDL language. Once this is completed, Verilog coverage should be evaluated, and any required extensions made to the information model. This process will clearly identify where the information between the two languages is the same and where it is different. This approach also will facilitate adding new common information to the information model (and, only if necessary, in the two languages), such as a common constraint language, or possibly analog extensions, or support for higher architectural levels of design. To the extent that the information model between VHDL and Verilog overlap (i.e., the information being modeled is the same information), it should be observed that the PI to access that information is identical.

In the synthesis area, to support the concept of raising synthesis to a higher level, it is recommended that a standard set of synthesizable primitives among HDLs be established.

To address HDL independent design analysis, a promising approach is to complete the work of the OMF, whose goal is to have a simulator interface for compiled HDL models (be they VHDL, Verilog, C), and for the various languages to have a compiler to that OMF interface. This would support the concept of language neutral standard interface for simulators.

It is recommended that the work of the OMF be accelerated, that "negative delay" as proposed for VITAL 3.0 be supported at initial release of the OMF, and that an HDL-independent simulator interface be developed to meet the requirements of the design scenarios described above. It is further recommended that the planned OMF interface be later extended to include analog support (after VHDL-A and VERILOG-A are balloted).

Currently, the OMF effort is only targeted for HDL independence for simulation. Synthesis and formal verification tools must continue to support multiple HDLs until and unless additional approaches beyond the planned OMF become visible which could make those design tools more HDL independent.

For meeting the requirements of constraint driven design, it is recommended that standard keywords be established, usable in both VHDL, VERILOG, and any other design language, to specify target values. This would most likely be an ongoing collection of keywords much like a message dictionary. CFI could be the keeper of this standard.

It is also recommended that a standard be developed to support the floorplanning requirements described above.

5.3.4 Roadmap - System Level Design

5.3.4.1 Standards to Support System Level Design

It is recommended that the Industry Council commission a task group to begin work on a system design standard. As used in this document, system level design includes both architectural and traditional high level design. Innovations in architectural design are emerging for which new standards will be required. Since VHDL is significantly more comprehensive as a design language for higher level system design, the recommended approach is to start with VHDL, and build an information model. Then, information model coverage of Verilog should be tested (with the assumption that much of the information model derived from VHDL also has a relationship to Verilog). It is the goal that once these information models are developed (and extended for architectural design as required), a system design standard can be released. Potentially, use models for VHDL and Verilog could also be released.

This item is high priority for the near term.

5.3.4.2 HDL Standards to Support Synthesis

A standard set of synthesizable primitives among HDLs must be established. An IEEE work-group is currently working on the problem of synthesizable subsets. The Industry Council should endorse this effort or commission a task group to address this problem.

This item is high priority for the near term.

5.3.4.3 Standard HDL Interfaces to Design Analysis Tools

The OMF effort described above is high priority in the immediate time frame including the support for negative delay in constraint specification as well as in normal path delay specification.

The analog extension is high priority *after* VHDL and VERILOG support analog. This item should be implemented as high priority in the immediate time frame.

In this area, efforts must continue to focus on improved interoperability of models from different sources (suppliers) to support system design and analysis.

Strategically, this task should be extended to also address HDL independent fault simulation and test analysis of hierarchical, mixed-level system models.

5.3.4.4 Standard Controls for Constraint Driven Design

Standard keywords, usable in VHDL, VERILOG, and any other design language to specify target values for the metrics discussed above, must be established. It is recommended that the Industry Council commission a task group to focus on this work.

This item is high priority in the immediate time frame.

5.3.4.5 Standards to Support Floorplanning

A standard must be developed to support floorplanning requirements.

This item is high priority in the immediate time frame.

5.4 Detailed Design

This section addresses detailed design: the current environment, requirements, recommendations, and roadmap.

5.4.1 Current Environment - Detailed Design

In the detailed design phase, the system level design created in the previous step, is transformed into detailed logical and physical design levels for a given technology.

In the logical design phase, detailed logic design is created and analyzed. In selected situations with today's technology, the logic of certain elements can be completely synthesized into technology specific gate level designs. In other cases, manual entry of the gate level logic of the design is required, at least in part. In any case, verification of the functionality and timing of the detailed level of the design is performed in this phase. Also in this phase, given high level layout of logic, *estimations* of testability, and certain design quality and reliability, can be performed (e.g., thermal analysis, power estimation).

In the detailed physical design phase, the detailed physical layout of the design object takes place. The detailed placement and wiring interconnect of the logic of the design is accomplished, guided by the high level constraints and floorplanning steps from high level design, and detailed logic design. Next, an assessment of the impact of the placement and wiring interconnect on the overall timing of the design is performed (e.g., accurate timing analysis across design levels, hierarchical interconnect modeling, hierarchical parasitic extraction and modeling). Also in this phase, the *analysis and measurement* of design quality and reliability is performed (e.g., signal quality analysis, power grid analysis, thermal analysis, power analysis).

In the current environment there are a host of standard and defacto standards in use, including EDIF, CFI DR, PDEF, DEF, SPF, SDF, and several others. EDIF and CFI DR (and STEP AP210) are, to varying degrees, an attempt to address the needs of both the logical and the physical design data representation areas. However, there is no current adequate standard for logical connectivity that supports a file-based EDIF-like approach, as well as a programming interface DR-like approach, in a consistent way. In addition, there is no standard for chip physical design data, and the standard for MCM physical design data is not yet completed by EDIF. EDIF PCB/MCM was selected by the MCM ASEM alliance as the standard for MCM physical design data. CFI's current plans are to converge to a standard information model with EDIF (and eventually converge information models with STEP), so that the CFI programming interface evolves from a common information model.

5.4.2 Requirements - Detailed Design

Below are listed the key information requirements needed to support the detailed logical and physical design phases for all package levels.

5.4.2.1 Standard Detailed Design Representation

A standard is required for an integrated representation (i.e., a file format and a programming interface based upon a common information model) for all detailed logical and physical

design information and the interrelationships between them, including support for connectivity and related annotation of logical and physical design information. It must define a standard base from which to support all types of packages as defined in this document. It must include sufficient capability to support interoperability in a hierarchical and incremental design environment between all logical and physical design tools, as well as interoperability with the system design environment, with correlation to HDL design elements. It must provide an effective interface for manufacturing build and mechanical design. It must support parameterized connectivity to enable a single primitive to support a variable number of inputs.

To adequately support detailed logical and physical design, there is a need for a design representation that maintains the relationships between the logical and physical components and access points and the logical connectivity model. This enables interoperability between tools that extract parasitics, calculate delays, and that do back annotation, manufacturing diagnostics and repair actions, etc.

In Table 5.1, “Impact of Design Size on Design Processing Times,” on page 85, there are several major items of interest:

- The columns entitled “Transistors per Chip” and “ASIC Gates per Chip¹” describe the projected number of transistors and chip area over the next decade
- The column entitled “Relative Design Data Size Increase” uses 1995 as a base for normalization of data to show the *data explosion* for the years that follow:
 - 1995 is “1”; i.e., the point of normalization.
 - In 1998, the number of transistors nearly triples (from 5M to 14M). Therefore, whatever the design data size was in 1995, it will increase by a factor of 2.8 in 1998 (for ASICs), and so on. By the year 2010, the amount of design data will be 86 *times* the amount of data in 1995.
- The column entitled “Relative Simulation Times” uses a similar approach. Based on historical and empirical data, the amount of simulation time (e.g., CPU time) required to do simulation is n^2 times the number of circuits being simulated. Taking the design size information and squaring it yields the relative simulation time for that size of design (e.g., $2.8^2 = 7.8$). Again, by the year 2010, the projected simulation time for a design which is 86 times as large (for ASICs) as designs in 1995, will be 7396 *times* as long as it is in 1995. Simulation is only one of the design activities impacted by the tremendous growth in chip density; all design activities that tend to operate on “entire designs” will face this issue.

1. The National Technology Roadmap for Semiconductors, Semiconductor Industry Association, 1994, Overall Roadmap Technology Characteristics, Table 2 on page 16.

This data, based on the projections in the NTRS Roadmap, clearly indicates the importance of the hierarchical design approach and incremental processing, as required approaches to support the very large chip designs of the future. We must take action *now* to support hierarchical and incremental design representations, and the design tools of the future must be designed to handle the processing of such design representations.

Table 5.1: Impact of Design Size on Design Processing Times

Year	Transistors per Chip	ASIC Gates per Chip	Relative Design Data Size Increase	Relative Simulation Times N^2
1995 (0.35m) ASIC uP	12M	5M	1 1	1 1
1998 (0.25m) ASIC uP	28M	14M	2.8 2.3	7.8 5.3
2001 (0.18m) ASIC uP	64M	26M	5.2 5.3	27 28
2004 ASIC uP	150M	50M	10.0 12.5	100 156
2007 ASIC uP	350M	210M	42 29	1764 841
2010 ASIC uP	800M	430M	86 67	7396 4489

This key standard must also support all requirements stated in 5.1 "Common Topics Across Design Information", with special focus on 5.1.1 "Incremental Processing" and 5.1.2 "Hierarchical Processing".

5.4.2.2 Standard Detailed Design Representation - Extensions for PCB Packages

This package type has a number of unique requirements that must be addressed by any proposed standard for design and manufacturing release. The outline of a PCB is frequently odd shaped and designed in mechanical design. See 6 "Key Interfaces to Other Domains" for key manufacturing and mechanical design interface requirements.

In certain instances, raw boards are designed and manufactured to allow choices of components in selected locations. In addition, this level of package can be reworked. Components may be added/deleted; wire connections can be added and deleted after initial build. These choices may have different lead spacings requiring a common bond site pattern. There is a need to be able to identify what has changed in a design. The standard must support requirements unique to this type of package.

5.4.2.3 Standard Detailed Design Representation - Extensions for MCM Packages

There are several types of MCMs. The MCM type that has chip(s) sitting in a well has unique requirements. The standard must support requirements that are unique to this type of package.

5.4.2.4 Standard Detailed Design Representation - Extensions for Chips, Macro-Cells

5.4.2.4.1 Concurrent Design Chip Requirements

To provide for EDA support for concurrent front end and back end physical design automation and aids, chip physical information needs to be integrated with the rest of the hierarchical electronics design representation. Support is needed for tracking of timing and performance, requirements, and design decomposition, across multiple levels of design and diverging design hierarchies. The tracking information is important in accurately back annotating, later more accurate design details back up to higher level, and earlier versions of the design, as part of the re-verification process.

The chip physical representation needs to support implementation of engineering change order (ECO), and other incremental change requirements by efficiently handling of redesign through re-use of previous physical design. The representation needs to support library changes, net list and other high level design representation changes, placement adjustments, and routing edits.

5.4.2.4.2 Logical to Physical Correlation Requirements

GDS-II Stream format is the current defacto standard for physical design maskout information, however, it does not support the correlation to the logical connectivity model. For extraction tools to have access to the rest of the design information needed to calculate delays and parasitic information, the extraction tools must either re-derive the original net and circuit relationships, or use the original design data that is not part of the interface data. It is more accurate and faster to store these relationships in the design data base than to re-derive the relationships.

5.4.2.4.3 Design and Tool Sharing Between Engineering and Manufacturing

Engineering tools used to view the physical design, are also required in manufacturing. Checks for manufacturability should be run earlier in the design process in the design shop. The engineering design tools use the relationships between the physical shapes and the logical connectivity. This engineering/manufacturing interoperability is another important reason to save these relationships in the physical design data, both in engineering and across the engineering/manufacturing interface.

5.4.2.4.4 Requirements Unique to Chip Packages

The physical design information for the chip (macro-cell and full chip), is much more voluminous than for other package types. There is no concept of unpopulated chips, or any concept of “assembly data” as in other package types. In an ASIC chip, the physical data from the circuit library is merged with all the other design elements or cells via placement into one collection. However, the design tools and analysis tools still require the correlation to the original logical design connectivity and schematic data.

5.4.2.4.5 Form of Incremental Release

The ASIC chip has a “front end of line” and “back end of line” concept. The back end of line data is on the last few mask levels in the manufacture of the chip. This consists of a restricted set of shape data from the circuit library and the interconnecting wire data. The standard must support requirements, such as this, that are unique to the chip level package.

5.4.2.5 Placement Data

There is a requirement for a standard to provide a means to efficiently record placement of cells, cores, and components to support a standard interface that will be used between:

- placement tools and wiring interconnect tools.
- placement and delay estimation
- placement and timing driven synthesis.

This information will be passed along with and be correlated to the logical connectivity model, or may be passed as an incremental piece of information. Performance is a concern in tool interactions such as those listed above. In addition to the requirement for chip support, these concepts apply to higher level package types (i.e., boards, MCMs)

5.4.2.6 Standards to Support Floorplanning

There is a requirement to share information between the synthesis and initial floorplanning activities performed during system level design, and the detailed floorplanning with subsequent placement and routing for wiring interconnect.

This data includes the specification of information described in 5.3.2.4 "Standards to Support Floorplanning", and additional refinements to meet detailed floorplanning requirements. This information must also be shared with the system level design phase so that high level floorplanning, timing analysis, and synthesis can be effectively linked with detailed floorplanning, placement and wiring, etc. Support for incremental and hierarchical processing is required in these activities.

5.4.2.7 Standard Language for Chip Layout Generators

Physical layout generators should use a language that is standard and universal. Just as most computers now have a language compiler that supports C for portability across operating system platforms, layout generators need a standard language to help with portability of the layout across different design groups and different toolsets.

The rationale for this requirement is to ensure that layouts can be re-used (or regenerated) by future layout programs as time elapses and design systems evolve. The longevity of IC layout designs can be better achieved through the use of a standard IC layout generation language. Pressures to achieve better design re-use, and portability of layouts, will only increase the desire to capture physical layout in standard tool-neutral formats.

5.4.2.8 Standards to Support Testability Analysis

5.4.2.8.1 Standards to Support Manufacturing Test Rules

There is a requirement for a standard method to describe the rules of a manufacturing test process, so that checking against those rules can be performed during the design process. These manufacturing process capability rules are required for all levels of packaging, to enable design for testability to be part of the design process.

Detailed testability analysis must be run early in the detailed design cycle, and as part of the final manufacturing test data generation. Subsets of testability analysis, are even run as part of the synthesis process. No matter when it is run, there is a need for the following:

- a manufacturing fault model (e.g., for traditional stuck fault coverage)
- delay fault model (for designs that are pushing technology performance).

These test rules are required to support testability analysis during the design process. When the design is pushing the technology limits for performance, then delay test generation, which requires a delay fault mode, is run.

5.4.2.8.2 Support for Test Vector Specification

In the process of performing testability analysis or final test data generation as described above, test vectors may be generated and captured. Any standard for test vectors must support the specification of this test vector information. This information is part of the key interface to manufacturing and is discussed further in 6.2 "Manufacturing Test Interface".

5.4.2.8.3 Standards to Support Component Self-Test

As circuit densities increase, use of test technologies that depend on inserted test logic (e.g., on-chip), will also increase. Automatic insertion of standard BIST and emerging test logic insertion must be supported in EDA Design tools. JTAG must be appropriately supported.

5.4.2.8.4 Standards for Component Test Data Re-Use

As components and cores (subsets of chips), continue to be re-used in new designs, the test information for those reusable design objects must also be captured, so the test data can be re-used in the design and test of the new design. This also applies to components that are used in higher levels of package, so that the component test data can be used in testing the higher level package. This test information must be included in the appropriate re-use library.

The standards for test data re-use must address test for reusable chip components, including cores. Two different kinds of test information are required, including:

- Information to support test of the component (or core)

This first requirement is satisfied by the appropriate conditions to enable controllability/observability of the embedded component, so that existing patterns can be applied, using an in-circuit type of test strategy.

- Information to support test of the circuitry in which the component is embedded

This second requirement calls for a standard that records modes of the component or core that allow a ‘flush-through’ mode, so that patterns for the logic around the component can be flushed through the component when certain control conditions are met.

That is, this requirement documents *test-related* (as opposed to functional) behavior of the component. Armed with this kind of information an ASIC designer would know how to configure the component or core to propagate signals through the component, so that the surrounding circuitry may be tested.

5.4.2.9 Standards to Support Manufacturability Analysis

5.4.2.9.1 Standard Support for Manufacturing Build Rules

There is a requirement for a standard method for describing the rules of a manufacturing build process, so that design/manufacturing rules checking can be performed during the design process. These manufacturing process capability rules are required for all levels of packaging to enable design for manufacturability to be part of the design process.

5.4.2.9.2 Support for Virtual Manufacturing

Currently, there are key steps in the design of electronic packages to verify the functionality and timing of designs. This concept needs to be extended to include a virtual manufacturing process capability, in the EDA Design System, so that the manufacture of the device (i.e., macro-cell, chip, or board) can be simulated, as well as the functional operation of the device.

5.4.3 Recommendations - Detailed Design

Action must be taken now to drive towards a converged and common core information model from which future standards effort(s) in design representation can be based. EDIF and CFI DR have recently designed a common core information model base for logical connectivity. This work should be accelerated and be expanded if necessary to include any other applicable standards. Over time, other industry standards related to detailed design must be included in this convergence strategy.

The goal must be to develop a common information model from which a file format (e.g., extended EDIF), and a programming interface (e.g., extended CFI DR) can be developed. Once a common information model is agreed upon, all future releases of the standard for detailed design, should include a synchronized and simultaneous release of both the file format and the programming interface for the standard. This approach, will ensure that both the file format approach and the programming interface approach, are developed from a common information model base, and that either approach (or both) can be used to maximize data interoperability across and within EDA design systems.

The priority for this convergence should be for logical connectivity first (which applies to all levels of packaging), then support for detailed physical design. Efforts for physical design should be for chip packages first, then physical design for boards (based on the EDIF 3.5.0 work). Every effort must be made to develop the physical design representations for chips, and for PCBs and MCMs in parallel, and as soon as possible.

5.4.4 Roadmap - Detailed Design

5.4.4.1 Converged Industry Standard for Logical Connectivity

The EDIF and CFI DR effort to develop a common core information model must be accelerated and completed as soon as possible. This information model primarily addresses logical connectivity. All detailed design industry standards that relate to logical connectivity must be part of this convergence effort, so that an industry-wide information model results from this work.

Based on the common core information model from the above effort, a new connectivity standard must be developed, that includes both a file format and a programming interface.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.2 Converged Industry Standard for Board Packages

Based on the above logical connectivity standard, extensions must be made to support the design and manufacturing build interface for high level board packages (i.e., PCA/PCB).

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.3 Converged Industry Standard for MCM Packages

Based on the above logical connectivity standard, extensions must be made to support the design and manufacturing build interface for MCMs.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.4 Converged Industry Standard for Chip Packages

Based on the above standards, extensions must be made to support the design and manufacturing build for chip subsets (e.g., macrocells) and chips. This standard must include support for placement information as described in 5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells".

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.5 Placement Data

Define a standard to efficiently record placement of cells, cores, and components, that will be passed along with and be correlated to the logical connectivity model, or may be passed as an incremental piece of information.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.6 Standards to Support Floorplanning

Define a standard to support floorplanning extensions required to support the interface between detailed floorplanning, placement, and routing for wiring interconnect.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.7 Standard Language for Chip Layout Generators

Define standards for IC layout generation.

This item should be adopted as medium priority immediately, i.e., in the short (immediate) term.

5.4.4.8 Standards to Support Testability Analysis

5.4.4.8.1 Standards to Support Manufacturing Test Rules

Define a standard for Manufacturing Test Rules, to include traditional stuck fault models and delay fault models.

This item should be adopted as high priority immediately, i.e., in the near (immediate) term.

5.4.4.8.2 Support for Test Vector Specification

Define a standard for test vector specification.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.8.3 Standards to Support Component Self-Test

Define standards required to support insertion of BIST and other test logic.

This item should be adopted as medium priority immediately, i.e., in the near term.

5.4.4.8.4 Standards for Component Test Data Re-Use

Define standards required to support component test data re-use including:

- information to support test of the component (or core)
- information to support test of the circuitry in which the component is embedded.

This item should be adopted as medium priority immediately (i.e., in the short (immediate) term.)

5.4.4.9 Standards to Support Manufacturability Analysis

5.4.4.9.1 Standard Support for Manufacturing Build Rules

Define a standard for specifying rules for the manufacturing build process, for all levels of packaging, to enable design for manufacturability.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.5 Design and Technology Re-use

This section discusses the environment, requirements, recommendations and roadmap in support of design and technology re-use.

The EDA System Integration and Interoperability (EII) and Technology and Library Models (TLM) Working Groups both addressed standards issues in this area, and the subsections which follow represents a merging of the results of those groups. The TLM Working Group focused on support of the design process and the creation and maintenance of the required models, and the libraries for those models. A key purpose of their effort was to determine a baseline of existing standards for required models and to make recommendations for Roadmap action.

The approach taken by the working groups will help to maintain the Roadmap as design methods, technologies, and standards evolve. The approach of the TLM group allows the review of design and technology model re-use from a design system (infrastructure and tools) and the design information (design data representation) viewpoint. The proposed methodology was designed to expose current standards conditions and to aid in identifying areas where future standards efforts are required. Additional results of the TLM working group are available separately on the CFI ftp server.¹ The information in that report is divided into three parts; (1) Model Data Representation Standards, (2) Model Content Standards, and (3) Access to data and models in libraries.

Within the first area above, a table presentation of design views (or steps in a typical design process) versus technologies (integrated circuits, PCBs, MCMs, etc.) provides a base for evaluation of standards related to Model Representation Standards (Table 1). Each cell in the table enables a review of the applicable standards, the priorities associated with each design step, and other information which could lead to additional Roadmap recommendations. The Model Content Standards table (Table 2) has the same cells as the first table, but provides information on model accuracy, completeness, and content quality, as well as industry priority and importance, and other related standards information (such as extensions or enhancements required). Today's models do not address this content quality issue. Table 3 contains standards information that deal with model access, storage, nomenclature, interfaces, exchange, classification, maintenance, and validation.

Other information from the TLM report as well as input from the EII Working Group has been embedded in the appropriate sections below.

1. Status of the Activities of the Technology Libraries and Models Working Group, 7/30/95, available in PostScript form on the CFI ftp server (ftp.cfi.org) at /public/Cfi/Development/Roadmap/TLM/Matrix6.ps.

5.5.1 Environment - Design and Technology Re-Use

The National Technology Roadmap for Semiconductors states "new approaches must be found if industry is to continue on the 30% per-year, per-function cost reduction trend". Over the past few years, we've seen the development and exploitation of different technologies and methodologies to maintain this growth curve. One of these technologies includes using higher level design techniques, particularly languages such as VHDL and Verilog. Combined with design synthesis, this provided a great leap forward in representing design intent. Additional innovations in the very front end of design, however, are required and anticipated.

Incremental and hierarchical design techniques have also become increasingly important as we move up the complexity curve. Breaking a design up into manageable segments allows us to put more total engineering resources into the design, and exploit concurrent software and hardware (and eventually mechanical) co-design techniques.

Data management, in terms of information and complexity management, continues to grow in importance. No longer does a small group of engineers start and finish the entire design, self-contained, in their organizational cube. A single chip design today might be a massive system undertaking, with hundreds of engineers spread across global and corporate boundaries. Software management techniques for controlling the engineering data allows us to maintain our senses, without completely tripping over each other in the complex design.

Ever changing business practices in the '90s have enabled us to be more productive. We no longer try to do it ourselves. Who would have ever thought in the '80s you would see a joint chip designed by Apple and IBM? Or that TI and Hitachi could have ended up co-funding a new wafer fabrication facility for memory designs (Twinstar, Inc.). In today's business environment, we know we cannot do it ourselves. So, how can a design engineer take advantage of this global environment?

One answer is through design re-use.¹

Why Re-Use?

Re-use increases designer productivity. To begin with, there is less reinvention of the same thing. From a resource perspective, re-use allows incredible leveraging of prior work.

Re-use, from a designer perspective:

- enables others to independently develop building blocks in their area of expertise,
- enables leverage of prior work to produce new products using pre-designed building blocks, and
- enables added value to be provided for your customers, at reduced cost and with a shorter time to market.

By exploiting design re-use of existing or standard components, you can focus on more advanced technology and leading edge products for your customers, instead of spending time and money on something that has been done before.

1. "Enabling Re-use Provides Product Design Productivity", George Chandler (Texas Instruments), Sumit Dasgupta (IBM/SEMATECH), Gary Panzer (Hughes Aircraft Company), 8/95.

Design Re-Use

Re-use has been used for years in other industries, from automotive to software. Once the Ford 302 cubic inch engine block was designed, it was used repeatedly in many different car styles for years to come. For years, re-use has existed at the component level; we already have an existing business model -- the standard component data books -- such as the TI TTL data book. Every engineer had one and used it to search for an appropriate component to use in his or her design. Re-use was easy. The function of each entity was generic, but limited in complexity.

Re-use did manage to work its way into the chip level design and founded a type of design methodology called “standard cells.” But there is still more to gain through wider usage of macrocells and functional blocks. Furthermore, there is still potential re-use at different levels of design and abstractions.

5.5.2 Requirements - Design and Technology Re-Use

Before identifying and reviewing the key requirements for design and technology re-use, some definitions are in order.

Levels of Re-use

For the purposes of this document there are several kinds of reusable design objects.

1) Reusable Design Object Specification

This specification is comprised of many “*datasheet*” or *design object specification* types of information about a reusable object. Examples of the information in these specifications are shown in the list below. This information base is the initial access point for the library of reusable design objects and is used to help designers determine if a reusable design object exists which can be reused in a new design situation. Typically, a Design Object Specification would exist if, and only if, one of the reusable function or component design objects also exists (see the definitions that follow below).

Only by knowing what the original designer intended will the person using the design object (the “re-user”) have confidence in it. A certain level of knowledge is needed for confidence. The following list are representative elements of knowledge we’ve found are most necessary. For each one, as it applies to each level of design (macro, chip, and system), the units may be different, but the basic concept applies.

- price
- time to market/cycle time
- performance
- function
- reliability/availability/serviceability/manufacturability
- power
- quantity

- technology
- physical design
- environment
- testability.

For example, a different unit may apply for various uses of the knowledge element “price”. For a macro level, it may be simply the intellectual property price of using that element. For a system, it may be simply the unit price.

Every reusable design object (e.g., component, function, etc.) must have a Design Object Specification so the information can be accessed by appropriate search and retrieve methods.

2) Reusable Function

A function design object refers to the concept of a *simulatable design specification for the design object* that can be reused. The examples here might start with ALU chips or entire microprocessor designs for which a VHDL or Verilog simulation model is available. Several types of simulation models are in fact desired as are described below in 5.5.2.3 "Standards for Reusable Functions". This type of reusable design object is used to enable simulation of a candidate object in a new design context or to develop a new physical technology implementation of a previously designed function.

3) Reusable Component

A reusable component design object refers to the type of *component for which a physical implementation exists* that can be re-used. Examples are ALU chips, macrocells that can be embedded in larger chips, etc. These designs have previously completed physical implementations in given technologies that are largely predesigned from a physical design perspective.

Limited parameterization may be possible. There would typically exist technology libraries from one or more suppliers of such reusable design objects.

Standards Promote Re-use

Re-use can enable productivity across a wide range of the design cycle, from architecture design all the way to fabrication. However, each phase of the design cycle requires knowledge to be passed from the original designer to the “re-user”. This knowledge needs to be passed in a data format understandable to the creator and the re-user. Only through standards can this knowledge be confidently moved from user to user, through time, and across systems.

In the meantime, the current set of standards needs to be examined as a whole, not as individual parts. This set should be pruned where need be and grown/updated where holes exist so the entire set can lead to an information model for design re-use. This will also lead to the slight lagging of supporting library sets, and in the near-term, actual knowledge libraries.

Key Requirements for Design and Technology Re-Use

The following are the key requirements for design and technology re-use.

5.5.2.1 Standards for Reusable Design Object Classification Hierarchy

To make any standards reusable, a standard “dictionary of terms” or classification hierarchy must first be developed. Standard terms (reference the CFI EDB Data Dictionary and Pinnacles efforts) are necessary to enable search and retrieval engines to “find” the reusable objects in libraries. Each of the areas of requirements below also add to the requirement for a standard design object classification hierarchy, and to the information contained in the library for the objects that are present.

This item is high priority in the short term.

5.5.2.2 Standards for Reusable Design Object Specifications

In order for a design object to be re-used, there are key pieces of design intent (i.e., knowledge) as well as other information described in 1) Reusable Design Object Specification above. The minimum information required for design object re-use must be determined and a standard dictionary of terms and a specification of reusable design object content are required for this Reusable Design Object Specification.

This item must be addressed with high priority in the short term.

5.5.2.3 Standards for Reusable Functions

Simulatable HDL models must exist for a reusable function. Examples are VHDL, Verilog, and potentially other simulatable design descriptions. There are a host of different applications for such simulation models including architectural, performance, RTL level, and other simulatable models. Some or all of these models are required to enable effective re-use of the function. Standards are required in this area to define a standard dictionary of terms and a taxonomy to support the classification of reusable functions.

This item must be addressed with high priority in the short term.

5.5.2.4 Standards for Reusable Components

Technology implementations (i.e., detailed physical designs) of more complex design objects are rapidly emerging, and there are no standards for their representation. Examples, such as macrocells (cores) and entire microprocessors, are candidates for reusable components. Standards are required to enable standardized access and re-use of such designs. Limited parameterization with some reusable components (e.g., bit width) is also a requirement.

This item must be addressed with high priority in the short (immediate) term.

5.5.2.5 Libraries of Reusable Design Objects

This requirement is to ensure the appropriate libraries of reusable components are put into place and they are based on the appropriate set of standards. The required libraries include the appropriate design object classification hierarchy and design object specification information described above, as well as entries in the appropriate function and component libraries.

This item must be addressed with high priority in the short (immediate) term.

5.5.3 Recommendations - Design and Technology Re-Use

The Industry Council should commission a task group as soon as possible to determine the base information for which standardization is required for *all* areas of re-use. This group should develop and expand upon this base work to enable the creation of standards for:

- Reusable Design Object Classification Hierarchy (Standardization of Dictionary)

For this item, the CFI Electronic Data Book (EDB) and the PINNACLES work offers a base from which this definition of terms can evolve.

- Reusable Design Object Specifications

In this area, the design object classification hierarchy defined above would be used to support the development of standards to support the interface to standards based search and retrieval engines which enable engineers to “find and obtain access to” reusable objects which meet their needs.

- Reusable Function

Initially, reusable models in this category would be supported by the classification hierarchy and design object specification work above. Initial library members would consist of the existing and legacy functional simulation models described in VHDL and Verilog, etc. Over the long term, additional model types and classifications might well evolve to meet emerging requirements.

- Reusable Components.

Finally, reusable physical component library members would consist of the existing legacy physical design information. Based on the design object classification hierarchy and design object specification efforts described above, additional physical design or component library types and classifications will evolve to meet technology requirements.

5.5.4 Roadmap - Design and Technology Re-Use

Re-use of information, models, and libraries is the key to increase designer productivity, leverage technology cheaply, and evolve an interchangeable set of standardized knowledge (knowledge libraries).

5.5.4.1 Standards for Reusable Design Object Classification Hierarchy

The Industry Council should commission a task group as soon as possible to determine the base information for which standardization is required. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy. Some progress has been made in this area in the RASSP program and that work should be considered by this task group. In addition, the work of the IEEE and Reuse Library Interoperability Group (RIG)¹ on software reusability should also be considered by this task group.

This task is high priority in the short term.

1. Standard Reuse Library Basis Data Interoperability Model (BIDM), RIG Proposed Standard RPS-001 (1993), Approved 4/1/93, Revised 1/3/95.

5.5.4.2 Standards for Access to Reusable Design Object Specifications

The Industry Council should commission a task group as soon as possible to determine the base information required to support a standard interface for the “search and retrieval” of design objects. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above.

This task is high priority in the short term.

5.5.4.3 Standards for Reusable Design Object Content

Standards must be developed to support the concept of reusable design object quality. These standards must address the issues of accuracy and completeness for a given use (e.g., performance models for high speed simulation, versus detailed gate level representation models for very detailed simulation and analysis). Such standards can be very useful to validate that models are equivalent and meet the same criteria. The concept of standards for content apply to all reusable design objects, certainly including reusable functions and reusable components as discussed below.

5.5.4.4 Standards for Reusable Functions

In this category, the first order of business must focus on the various simulation requirements to support reusable functions. As documented in 5.3 "System Level Design", high level simulation models at various levels are extremely important during the early phases of design, where hardware/software trade-off analysis and overall system performance are evaluated.

Standards for creating such reusable simulation models are needed to support:

- Architecture Design
- Performance Modeling
- Algorithmic Design
- Functional Design
- Technology Mapping

Again, the Industry Council should commission a task group as soon as possible to determine the base information required to support a standard taxonomy and dictionary of terms in this area to provide a base to develop and classify reusable functions. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above. The work done in the RASSP program to define a taxonomy for simulation models should be considered by this task group.

This item is high priority in the short term.

5.5.4.5 Standards for Reusable Components (Technology Implementations)

In the area of detailed design, both logical and physical design standards are required to support the concept of reusable components. Again, as documented in 5.4 "Detailed Design", detailed logical and physical information in standard form is required to enable reuse at the component (physical) level.

Technologies which must be supported in these libraries include:

- Discrete Components (e.g., transistors, resistors, etc.)
- Integrated Circuits
 - Custom ICs
 - DSPs
 - ASIC Cells
 - ASIC Cores
 - FPGA
 - PLD/CPLD
 - Memory (RAM, SRAM, DRAM)
- Assemblies
 - PCB (Digital, Analog, Mixed Signal, RF, Multilayer, Flexible)
 - MCM (MCM-Laminated/Deposited/Ceramic (Digital, Analog, Mixed Signal, RF...))
- System

Information in the detailed design category for reusable component libraries must include:

- Technology Mapping (to logic design)
- Logic Verification & Analysis
- Electrical Simulation & Analysis
- Test
- Physical Design
- Verification/Rule Checking
- Parasitic Analysis/Simulation

In addition, the category of reusable component information includes the specific information needed to support the release of the necessary manufacturing build information, including:

- Design to Manufacturing
- Fabrication
- Configuration Management

Refer to 6.1 "Manufacturing Build Interface" for additional information.

Similarly, in this category, the Industry Council should commission a task group as soon as possible to determine the base information required to support a standard taxonomy and dictionary of terms to provide a base to develop and classify reusable components. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above.

This item is high priority in the short term.

5.5.4.6 Libraries of Reusable Design Objects

Each reusable design object (e.g., reusable function or component) must have a Design Object Specification. This information is required so that the information which makes those objects easily found by search and retrieval engines which are compliant to the standard Design Object Classification Hierarchy. This design object specification must include *many* “datasheet” or design object specification types of information such those listed in 5.5.2 "Requirements - Design and Technology Re-Use" above.

Libraries of reusable design objects of the types described in this section must be built to the standards described in this section to make them accessible by compliant search and retrieval mechanisms. In order to capitalize on the significant productivity increases offered by design reuse, we must be able to “find and get access to” reusable design elements.

We must also be able to coexist with the legacy libraries and their contents while we migrate to a longer term more “reusable” library strategy as described in this section. The strategies described in 2.1.4 "Design Information Roadmaps" and in Figure 4.2— "Open EDA Data Interoperability Architecture" overviews a strategy for how that can be enabled.

This task is high priority in the short term.

6 Key Interfaces to Other Domains

This chapter contains information about key interfaces to other domains related to electronic design and test. The chapter is divided into four key areas; manufacturing build interface, manufacturing test interface, mechanical design interface, and software design interface.

An electronic device evolves into an end-product through various stages. The point at which the product gets “passed” to the manufacture or test engineer is discussed in this section. The interfaces to manufacturing, assembly and test are very important. Traditionally, a large percentage of the development effort is expended in this area. Clean and efficient interfaces are required to support the entire product life-cycle.

As described earlier in this document, a number of data representation standards will be used, enhanced, or developed to support the product design process. It is incumbent on manufacturing and test to use the information provided by the design process. The development of information models will bridge the gap between the traditional roles of design, manufacturing and test. A common understanding of the information being shared will help to support the emergence of improved concurrency in all phases of the design process.

6.1 Manufacturing Build Interface

This topic discusses the data requirements that manufacturing needs from product development in order to successfully build the product. This includes data for manufacture of the bare die in the case of chips, or data for board assembly and test.

6.1.1 Current Environment - Manufacturing Build Interface

At various points in time, there is a need for information exchange and sharing between the development and manufacturing organizations. This phase is primarily characterized by the “release” of the final design of the design object to a manufacturing organization for the purpose of performing the manufacture of the product. However, in order to facilitate effective manufacturability of the product, it is also important for manufacturing to be involved early in the design process. Through “early manufacturing involvement (EMI)”, design for manufacture can be part of the process at each of the design phases.

The increasing use of re-usable chip subsets (cores) present a new test challenge. Designers are often forced to tie the core interface to I/O pads in order to gain access both to the core itself and to the circuitry where the core is embedded. This design technique is unsatisfactory for submicron design features. DFT techniques will have to be applied to cores, with corresponding documentation requirements (see 6.2 “Manufacturing Test Interface” for additional information).

6.1.2 Requirements - Manufacturing Build Interface

Manufacture (and test) engineers need information on product design, tooling, tolerances, fabrication sequences, dimensional requirements for design features, bare die and bare board testing information. Board assemblers need information on components, point of origin, assembly sequences, board aide relationships, location of fiducials, and electrical test vector information for in-circuit or functional testing. This information needs to be captured during the design process, even though much of this information is manufacturing build or test specific. The goal of the manufacturing and/or test engineer is to ensure that the information they need to complete their job is provided in a timely and efficient manner.

6.1.2.1 Develop Design Representation Standards to Include Manufacturing Information

There are manufacturing-specific information requirements that are not currently covered by existing design representation standards. This information, which is a part of the total product description, needs to be added to the information model and subsequent standards developed for chips, PCBs, and MCMs. In the area of PCBs, considerable work has been accomplished by IPC in the IPC- D35x series of standards for PCBs. Now, the design information must be integrated with the requirements of CAM systems.

6.1.2.2 Standard Support for Manufacturing Build Specification

There is a requirement for standard manufacturing build specifications for all levels of packaging including manufacturable chip subsets (macrocells) and entire chips, MCMs, and boards (PCA/PCB). A standard interface for passing all manufacturing data that is available from the design process into manufacturing is required (note that not all the data that manufacturing needs is available during the design phase).

This interface must include the support for logical to physical relationships discussed in 5.4 "Detailed Design". Refer to that section for additional information on key manufacturing build interfaces such as

5.4.2.2 "Standard Detailed Design Representation - Extensions for PCB Packages"

5.4.2.3 "Standard Detailed Design Representation - Extensions for MCM Packages"

5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells"

6.1.3 Recommendations - Manufacturing Build Interface

Action must be taken now to drive towards a converged and common core information model from which future manufacturing interface standards effort(s) in design build representation can be based. The work described in 5.4 "Detailed Design" should therefore be accelerated.

Over time, other industry standards related to the design-manufacturing interface must be included in this convergence strategy. As soon as possible, every effort must be made to develop the physical design representations suitable for manufacture of PCBs, MCMs, and chips in parallel.

These design and manufacturing interface standards should be integrated and extended to provide complete coverage of all key manufacturing interface requirements. Note that we may end up with multiple standards (use models built from a common information model) addressing the same kind of information but for different package types.

The standard manufacturing build interface requirements must be a convergence of standards that are currently under development (including EDIF PCM/MCM, CFI DR, and STEP AP2xx, as well as the proposed converged test information model). These efforts should be tracked to ensure that all key requirements are satisfied by the standards. This provides an opportunity to develop a single set of standards for the “manufacturing-to-design” information flow.

6.1.4 Roadmap - Manufacturing Build Interface

The roadmap for manufacturing build is to properly design the data representations for each package to include the necessary manufacturing information.

6.1.4.1 Standard Manufacturing Build Interface for Chips (and Macrocells)

Refer to section 5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells" for details on this standard.

This item should be addressed as high priority in the short term.

6.1.4.2 Standard Manufacturing Build Interface for PCBs

Refer to section 5.4.2.2 "Standard Detailed Design Representation - Extensions for PCB Packages" for details on this standard.

This item should be addressed as high priority in the short term.

6.1.4.3 Standard Manufacturing Build Interface for MCMs

Refer to section 5.4.2.3 "Standard Detailed Design Representation - Extensions for MCM Packages" for details on this standard.

This item should be addressed as high priority in the short term.

6.2 Manufacturing Test Interface

This section addresses the interface to ASIC, MCM, and PCB test. Tests include bare-die and bare-board, in-circuit, functional, and diagnostic tests.

6.2.1 Current Environment - Manufacturing Test Interface

Design-for-test (e.g., design for testability, fault modeling/analysis and grading, ATPG techniques, insertion of low-overhead BIST) is frequently addressed late in the cycle after the architectural, high level, and detailed level design has been completed. A key to success in the designs of tomorrow will be to consider design-for-test as early as possible in the design cycle. In addition, evolving test methods such as self-test must be addressed.

A major impediment to new design-for-test processes is the lack of any standard representation for test information. Designers are unable to express test strategies or configurations in a way that can be understood by many tools. Designers are therefore limited to single-tool solutions for test. The lack of a fully-developed and elaborated test standard further increases product cost and development time when outside components, including cores, bare die, and packaged ICs, are inserted into a design. If test information, especially design-for-test information, can accompany these components, then tests for the new design can be developed more efficiently. However, the transmission of test information for these components will not be successful in the absence of test standards.

As discussed in 3.1 "Emerging Paradigm Shifts", industry is moving to integrate the product development processes. An example of this, which is directly related to the test-related functions of CAD systems, is the test coverage management practice being developed on the RASSP program. Better integration of design and test information is important if the design and test processes are to be concurrent.

6.2.2 Requirements - Manufacturing Test Interface

Manufacturing test engineers have a need for much of the same information as that required for manufacturing build. In addition, there are some specific requirements such as those listed below. Refer to 5.2.2 "Simulation and Test Control" for additional information related to these requirements.

In addition, many of the requirements of manufacturing test are being driven earlier into the development process. Refer also to 5.4.2.8 "Standards to Support Testability Analysis" for test related requirements on the design process.

6.2.2.1 Standards for Mechanical Interfaces

The implementation of a manufacturing test requires enough physical information to allow a mechanical interface to the chip, board, or MCM under test.

For chips this information includes bond or probe pad size and location. The composition of the bond pad and access restrictions (e.g., no more than two touch-downs allowed) may also be required.

6.2.2.2 Standards for Electrical Interfaces

The implementation of manufacturing test requires enough electrical information to allow the application of test data to the chip, board, or MCM under test. This is especially important for digital tests, which are normally specified in terms of logical bit values. Analog tests are generally specified directly in electrical terms and are not included in this category.

On the other hand, modern manufacturing test practice is moving away from functional tests based on mission functions and is moving towards "vectorless" test based on test opportunities created by the details of the implementation technology. This means that the test designer needs access to electrical and physical characteristics of the product, and of the technology family with which the product is implemented, at the earliest possible time.

Vectorless tests generally operate from netlists with simple electrical interface descriptions; therefore, this information should be supported in the test standard(s).

6.2.2.3 Standards for Digital Test Vectors Specification

There is a requirement for a standard form(s) of test vector specification which can be used on any manufacturing tester that is compliant to the standard. The specific requirement is to define how digital test vectors can be mapped from one format to another.

There is a large number of digital test vector standards, both formal and de facto. Formal standards include WAVES (IEEE Standard 1029.1), and DTIF (IEEE P1029.4). De facto standards include those owned by Summit Design and by Teradyne. Many companies have well-established internal formats.

The plethora of formats impedes communication between tools and the creation of new, specialized tools. A single format would solve this problem, but any attempt in the short term to impose a single test vector standard would be difficult to achieve because these different formats each solve a problem for a particular constituency, and the needs of those constituencies must be respected.

6.2.2.4 Standards for Digital Scan and BIT/BIST

There is a requirement for standard forms for representing digital scan, BIT, and BIST test information.

Scan test data differs from general digital test vectors because the timing is very simple and only a few pins are involved in the test. Scan test data can include an enormous amount of test data behind them. Further, much of the scan data is repetitious, as preambles for loading specific internal registers continuously reappear in the test.

Many existing test vector standards attempt to address the special characteristics of scan. The principal focus is to express scan vectors compactly. The Serial Vector Format addresses the problem in the context of IEEE Std. 1149.1 (Boundary Scan). WAVES is quite flexible in the compact representation of serial test vectors but has no formal connection to product design (whereas the Serial Vector Format is intimately coupled to 1149.1 through BSDI).

6.2.2.5 Standards for Analog and Mixed-Signal Test Data

An increasingly important requirement for expressing tests for analog and mixed-signal products exists. Mixed signal designs are found today on chips as well as the more common MCMs and PCBs. The only standards currently available for expressing this class of tests are contained within the ATLAS family, IEEE Std 416 (ATLAS) and IEEE Std 716 (C/ATLAS).

Current standards suffer from inflexibility, as new kinds of tests generally require non-standard extensions. Indeed, this was recognized in C/ATLAS with the provision of EXTEND semantics. Non-standard extensions are not portable, and are unsuitable for any but point solutions to the test data exchange problem.

6.2.2.6 Standards for Diagnostic Information

Standards are required for manufacturing processes that permit repair require diagnostic information along with the test information. If a test detects a defect, the diagnostic information identifies the proper repair action (sometimes the proper “repair” action is to scrap the product!).

Defective MCMs are routinely repaired rather than scrapped. Defective PCBs may be repaired depending on the economics of the board. If it contains many expensive components, then defects will be repaired. Even processes for which repair is uneconomical can benefit from diagnostic information, since defect identification can be useful in process control.

The current de facto standard for diagnostic information at the PCB level is backtrace. At the chip level fault dictionaries are used. No strategy yet dominates MCM diagnostics.

6.2.3 Recommendations - Manufacturing Test Interface

Industry requires interchange formats for test information, and must address the harmonization of existing standards-related activities. These include STEP, EDIF, CFI, and the IEEE. The coverage of these activities overlap, and harmonization is recommended. On the other hand, the total test domain is not well covered, and extensions are required.

The integration of test information into design information is critical. This integration will lead to better specification of tests by designers, partly because test development will occur within their native environment, and also because the integration will support testability analysis tools that designers will find useful. Finally, a lack of integration will lead to test specifications that are inconsistent with the actual chip, PCB, or MCM.

We should start by harmonizing standards for digital test vectors. There are several formal and de facto standards currently existing, most of which are quite useful. A new standard is not necessarily needed in the short term, but the existing standards may need improvement. It is strongly recommended that test vector formats be converged through a common, core information model. Convergence establishes a formal relationship between multiple standards without forcing any standard to actually change. Over time, based on a common core information model for test information, the potential for a meaningful new standard in this area increases.

Standards must be developed for expressing test information that is more complex than digital test vectors. Some work is already being done in the IEEE (under SCC20 and CS DASC). Here again, there is an opportunity to encourage a single set of standards with no overlap.

Standards to support reuse of test information need to be developed. This is similar to reusing hardware components. A library of reusable test information supporting the Roadmap standards for test will jump-start acceptance of the Roadmap standards.

To address requirements for component test data re-use, we can start by looking at existing, commercial formats that deal with in-circuit test. In an in-circuit test, the pins of components that are not being tested are placed in a high impedance state so as not to interfere with the test signals being applied to the component under test.

Refer to 5.4.2.8 "Standards to Support Testability Analysis" for additional discussion on these topics.

6.2.4 Roadmap - Manufacturing Test Interface

6.2.4.1 Standards for Mechanical Interfaces

Mechanical information is well-defined by the upcoming EDIF 4 0 0 for PCB and MCM layout. EDIF 4 0 0 should be extended to support chip layout (at least to handle bond pads). Bond-pad descriptions from the DIE standard should be integrated into EDIF 4 0 0 so that a single format can be used to convey mechanical information.

Acceptance of EDIF 4 0 0 for test and probe points for PCBs, MCMs, and extensions to chip layout is a short-term goal. Integration of DIE information should take place within the near term. We should use the IPC-350 family of standards as examples of information that is desirable for manufacturing test. Simple three-dimensional geometry, suitable for defining the approach path of a probe-to-probe point, should be included in the near-term information model.

In the long term, support for three-dimensional geometry may be required, for instance, to define the approach path of a probe to a probe point. STEP AP203 and AP210 are sources of information models.

6.2.4.2 Standards for Electrical Interfaces

The adoption of EIA proposal 3257 (IBIS) is recommended for defining electrical interfaces. However, extensions or annotations may be required to indicate preferred voltage and current values for individual tests. These extensions are best captured in a test information model.

Integration of IBIS with the logical connectivity of the chip, MCM, or PCB is critical. This integration could be as simple as annotating the ports of the logical connectivity with pointers to the appropriate IBIS port specification.

In the near term, the requirements of vectorless tests should be addressed. This may be difficult since this is a test technology that is currently evolving. However, in the short term, the connectivity information used by vectorless tests should be expressed in the data representation standards developed under this roadmap.

6.2.4.3 Standards for Digital Test Vectors

In the short term a core information model for digital test vectors should be developed. Appropriate mappings from the information model to existing popular standards and formats should be developed. The core connectivity information model must be consistent with the core test information model.

In the near term modifications to popular formats should be purposed in order to achieve some level of harmonization between them. A reduction in the number of test formats in the near term would be a result of marketplace down-sizing to a few standards or formats. This will help to determine the direction of standards in this area in the long term.

6.2.4.4 Standards for Digital Scan and BIT/BIST

For the near term, the Serial Vector Format for testing 1149.1 products should be supported. The IEEE P1450 effort should be considered with extension to extend the proposed standard to address scan, BIT, and BIST. The specification of any standard in this area must be compliant with the information model for digital test vectors.

6.2.4.5 Standards for Analog and Mixed-signal Test Data

In the near term, a core test information model, which includes analog and mixed signal data must be developed. The PAP-E model has been proposed as the starting point for this model. As the PAP-E program has demonstrated, if the test interface is standardized via an information model, it becomes less important to standardize on particular data formats (such as WAVES for test vectors) as long as bindings, or mapping models exist between the data formats and the information model.

In the long term it is more important to enhance the existing standards and develop new standards for data formats with an eye toward supporting lossless mappings to the standard information model.

6.2.4.6 Standards for Diagnostic Information

In the near term, the IEEE Std 1232 (AI-ESTATE) should be considered as the core information model for diagnostics. This standard and its proposed component standards address fault trees and diagnostic inference models. The adoption of a backtrace model should also be supported.

6.3 Mechanical Design Interface

The design of enclosures for circuit subassemblies is one area that is usually a separate work task from the design of chips, MCMs and board packages. In different product classes, the exact definition of “Mechanical Design” may vary, but areas such as strength, vibration, heat, and the form/fit requirements of assembly (and disassembly, for repair) are important.

6.3.1 Current Environment - Mechanical Design Interface

The interface between mechanical design and electronic design for higher level packages such as boards (including MCMs, PCAs, PCBs) is a important interface to support concurrent design.

The end-product of detailed circuit design can now be automatically transferred into the mechanical design system via a narrowband, file-format exchange using a short list of formats. The most error-free formats are direct conversion into the proprietary database format of the Mechanical Design System. While standards exist capable of representing most of this information, their implementation by both electrical and mechanical design tools is spotty and suffers quality problems. This is sufficient for a sequential design process where mechanical decisions can be made after electrical decisions. This is a problem area already today for high frequency and high power applications such as the transmit/receive modules used in electronically agile active arrays.

6.3.2 Requirements - Mechanical Design Interface

This section overviews the key requirements related to the interface to mechanical design.

6.3.2.1 Support for Electronic Design and Mechanical Design Interface

Standards are required to support the specification and integration into the design of

- Mechanical information in the electronic design process
- Corresponding electrical information in the mechanical design process.

At a minimum, both processes share knowledge of package outline, connector locations, mounting holes, restricted areas, cable sockets and SMA plugs. The electrical domain needs to know the relationship between the precise location of an instance of a component part number and the functional role the component plays, as well as the electrical connectivity integrating it into the larger functional circuit.

Looking into the future, wideband information sharing between electrical and mechanical design will be required, based on both technology and product trends. Increasing functional density and product intelligence, together with more use of wireless and mobile communication, mean that the collapsing of product design into a far more integrated, concurrent process will require a much tighter coupling of mechanical and electrical design in the future. In some cases, for example flexible wiring elements, mechanical design may be the lead discipline throughout component design.

6.3.3 Recommendations - Mechanical Design Interface

The history of the electronic design community is to construct standard design representations in a *connectivity-first* view. Thermal analysis and mechanical design require a *location-first* view of the product. To achieve a workable seamless concurrent integration with mechanical design, the core information model and design representations must be normalized such that the design database is a two-port memory, supporting access in connectivity-first *and* location-first perspectives with equally.

This should be accomplished by treating the mechanical designer as a valued customer in the evolution of product and design representation standards. The roadmap below details how to integrate this objective with the other objectives of the design infrastructure.

6.3.4 Roadmap - Mechanical Design Interface

This section outlines the roadmap for the interface to mechanical design.

6.3.4.1 Standards to Support Floorplanning

Standards for “rough-out” of PCB and MCM packages should address mechanical performance requirements and performance risk. Mechanical performance prognostics need to be developed for packaging technologies and used at the floor-planning stage to identify design cells requiring concurrent electrical/mechanical design. Practitioners from current high-risk design domains such as radar receive/transmit modules should be drawn into the floorplanning representation design process to get the advantage of their lessons learned.

6.3.4.2 Converged Industry Standard for Board Packages

The Mechanical Design community must be involved in this process as a customer. Efficient access to design information on the basis of location or volume from a solid modeling viewpoint must be supported. Thermal analysis, manufacturing assembly, and repair disassembly are important walk-through scenarios in the design and acceptance process for this representation.

(See reference 5.4.4.2 "Converged Industry Standard for Board Packages")

6.3.4.3 Converged Industry Standard for MCM Packages

Thermal performance prognostics must be developed for high-power package technologies.

The Mechanical Design community must also be involved in this process as a customer. Efficient access to design information on the basis of location or volume from a solid modeling viewpoint must be supported. Thermal analysis, manufacturing assembly, and repair disassembly are important walk-through scenarios in the design and acceptance process for this representation.

(See reference 5.4.4.3 "Converged Industry Standard for MCM Packages")

6.3.4.4 Converged Industry Standard for Chip Packages

Ensure that the design representation standard supports the appropriate creation and extraction of mechanical information as required in the construction of higher-level packages (above).

(See reference 5.4.4.4 "Converged Industry Standard for Chip Packages")

6.3.4.5 Placement Data

Leverage the information structures used by the mechanical community in defining this standard. Integrate this representation and its development into the above three topics.

(See reference 5.4.4.5 "Placement Data")

6.4 Software Design Interface (Hardware/Software Co-design)

6.4.1 Current Environment - Hardware/Software Co-Design

Given the increasing cost of designing hardware (due to the size of designs) and of software (since most complex electronic systems are software driven), the interface between software design and electronic design is becoming much more important than ever before. Many of the key innovations in the design of complex systems are centered around the interface of software and hardware components, sometimes called hardware/software co-design. For architectural and high level design, evaluation of early hardware designs using techniques such as performance modeling is an emerging technique. Similarly, virtual prototyping enables designers to maintain a consistent view of the hardware developed for the software (and vice-versa) throughout the design cycle, thus easing the integration task and reducing the frequency and cost of redesign. Co-simulation of the detailed hardware and software design is used to verify the correct system behavior before the hardware is fabricated.

Virtual prototyping is defined to be a comprehensive model of a system or component that models the external and internal temporal, data-value, functional, and structural combination of aspects for the system or component. A virtual prototype may be written at any level of abstraction. However, the most significant usage of the term, virtual prototypes, occurs at and below the network architectural level.

In practice, virtual prototyping is done to allow the hardware and software designers to maintain a consistent view of the system as they develop their respective portions of the design. The hardware designers see an updated representation of the software and its requirements as they work and the software designers have an up-to-date representation of the hardware and its capabilities to target their software. This greatly aids the integration task and reduces redesign.

Hardware/software co-design is the simultaneous consideration of hardware and software within a system design and includes the co-development and co-verification of hardware and software in the system (from Myers, Bard, and Schaming - HW/SW Co-design white paper, Oct 94). Currently, co-simulation is used to ensure that the designed system operates as intended, typically with a fully-detailed hardware and software model. This is a low level of abstraction.

Co-specification entails deriving specifications for hardware and software components of a system based on the requirements and system specification. Co-specification is currently not typically performed in a formal sense, but rather glossed over. This is at a high level of abstraction.

6.4.2 Requirements - Hardware/Software Co-Design

The importance of software within the system design task becomes more important over time as system developers demand increasing performance and flexibility. The need for increased automation to aid in the development of hardware, software, and the interface between hardware and software is acute, as software development is often the bottleneck to system design, thus dictating the schedule and cost of the overall system design.

Increasing size and complexity of designs dictates that the importance and difficulty of hardware/software co-design will continue to grow. The requirements below need to be addressed.

6.4.2.1 Standards for Simulatable Specifications

A standard to define system requirements and behavior via simulatable specifications is needed. Simulatable specifications should support hierarchical top-down design, include system behavior requirements, design constraints such as timing, size, power, and weight, and tests to ensure the conformance of a design to the requirements. The semantics of simulatable specifications must be precisely defined. Simulatable specifications must be flexible enough to support different design methodologies and implementations.

6.4.2.2 Standards for Hardware/Software Partitioning

The ways to partition tasks into hardware and software and to map software tasks to processors need to be addressed via recommended practices, standard heuristics (akin to using simulated annealing as a standard approach to placement and routing), or standard algorithms.

Determining a set of viable alternative design approaches based on a mix of hardware and software components is a key step in the design process that must be better understood.

6.4.2.3 Standard Support for Rapid Hardware/Software Design Evaluation

Standard libraries or practices for performance evaluation via simulation (as with VHDL), either numerically or analytically are needed to support rapid searching through the hardware/software design space.

6.4.2.4 Standard Interfaces for Modeling Hardware/Software

A standard interface or set of interfaces is needed to model the hardware as viewed by the software and the software as viewed by the hardware.

6.4.2.5 Standards to Support Virtual Prototyping

Standard practices for virtual prototyping are needed that support different levels of design detail.

6.4.2.6 Standards to Support Co-Simulation

Standards for co-simulation and testing are necessary to provide the capability of executing software on simulated hardware. Testing the hardware and software with a standard hardware/software equivalent of “test vectors” as in IEEE WAVES would help facilitate tool portability and interoperability.

6.4.3 Recommendations - Hardware/Software Co-Design

In view of the requirements discussed in the last section, the recommendations in this area include:

- Develop a taxonomy of terms and definitions, using the RASSP taxonomy as a starting point, focusing on the following classification of types of systems in the domain of standards:
 - real time, embedded, general purpose
 - shared, dedicated; dynamic or static scheduling
 - serial, parallel
- Develop standard definitions, requirements, and/or forms of simulatable specifications
- Develop partitioning techniques, algorithms, heuristics
- Develop modeling standards/techniques/libraries
- Develop standard hardware/software interface models

- Develop standards for virtual prototyping with consistent hardware/software views through various levels of design detail
- Develop standard means to verify correct behavior of hardware and software together.

6.4.4 Roadmap - Hardware/Software Co-Design

Standards and standard practices need to be developed for many of these areas. Some products and university research tools exist that could become de facto standards, although it is probably too early to consider them mature enough to be included as formal standards.

6.4.4.1 Focus on Standardizing the Taxonomy of Modeling

Develop a taxonomy of terms and definitions, using the RASSP taxonomy as a starting point, perhaps including the classification of types of systems to focus on in the domain of standards.

This task is medium priority for the near term.

This page was intentionally left blank.

Appendix A - The Roadmap Development Team

The following people from across the EDA industry and around the world participated in the development of the EDA Industry Standards Roadmap. The three working groups are shown with the working group chair(s), and working group champions or other key participants are identified within each group.

- EII - See Table A.1: "EDA System Interoperability and Integration Working Group (EII)"
- DDM - See Table A.2: "Design and Data Management Working Group (DDM)"
- TLM - See Table A.3: "Technology Libraries and Models Working Group (TLM)"

Table A.1:EDA System Interoperability and Integration Working Group (EII)

Name	Representing	Background	Participation
Grant Martin	Cadence	User	Co-Chair
John Teets	CFI	CAD Integrator	Co-Chair & Champion Design System
Elfriede Abel	ECSI Europe	Standards	Review
Malcolm Ash	Mentor Graphics	Developer	Review
Dieter Bergman	IPC	Standards	Review
Victor Berman	Cadence	Developer	Review
Dick Bushroe	HP/SEMATECH	CAD Integrator	Roadmap Sponsor
George Chandler	Texas Instruments	User	Co-Champion Reuse
Ron Christopher	Independent	CAD Integrator	Champion Design Information
Jim Clark	Electronic Tools Company	Developer	Review
Don Cottrell	CFI	Standards	Review
Sumit Dasgupta	IBM/SEMATECH	Developer	Assist Champions
Shaun Devlin	Ford	Standards	Review
Mike DonTigny	Mentor Graphics	Developer	Review
Peter Eirich	Westinghouse	Standards	Review
John Eurich	Engineering DataXpress	Developer	Review
Mark Falco	Lockheed Martin	CAD Integrator	Review
Gary Ferrari	TechCircuits	User	Review
Rob Fletcher	IEEE	Standards	Review
Steve Fortier	Intermetrics, Inc	Government	Champion Key Interfaces
Donna Fritz	EDAC	Standards	Review

Table A.1:EDA System Interoperability and Integration Working Group (EII)

Name	Representing	Background	Participation
Anthony Gadiant	PDES, Inc.	User	Review
Al Gilman	Intermetrics, Inc	Integrator	Review
Rita Glover	EDA Today	EDA Consultant	Review
Steve Grout	SEMATECH	User	Review
Tamotsu Hiwatashi	Toshiba	CAD Integrator	Review
Jan Johansson	Ericsson	User	Review Only
Hilary Kahn	University of Manchester	Standards	Review
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	Review
Bipin Chadha	Lockheed Martin ATL	CAD Integrator	Review
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
Mike McIlrath	MIT		Review
Larry Melling	IKOS	User	Review
John Mermet	ECSI Director	Standards	Review
Joe Morrison	Loral	User	Review
John Murphy	Cadence	Developer	Review
Mike Krause	Nortel	User	Review
Mika Nuotio	Ericsson	Developer	Review
Rick Ong	Motorola (RASSP)	CAD Integrator	Review
Gary Panzer	Hughes Aircraft/RASSP	User	Reuse Champion
Greg Peterson	Wright Patterson AFB	Government	Software I/F
Sishpal Rawat	Intel	User	Review
Patti Rusher	EIA/EDIF	Standards	Review
Lee Shombert	Intermetrics, Inc	Integrator	Review
Mike Tong	AT & T	CAD Integrator	Review
John Welsh	Lockheed Martin ATL	CAD Integrator	Review
Tom Vanderberge	Texas Instruments	User	Review

Table A.2: Design and Data Management Working Group (DDM)

Name	Representing	Background	Participation
Don Cottrell	CFI	Developer	Co-Chair
Danny Davis	Intermetrics	Developer	Review Only
Dieter Bergman	IPC	Standards	Review
Ron Christopher	Independent	CAD Integrator	
Robert Fletcher	IEEE	Standards	
Ted Frederick	Cadence	Developer	Co-Chair
A. J. Incorvaia	ViewLogic	Developer	
Donna Fritz	EDAC		
Rich Goldman	EDAC, Synopsys	Developer	Review
Steve Grout	SEMATECH	User	
Jan Johansson	Ericsson	User	Review Only
Sunil Joshi	Sun	User	Review Only
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
John McClintock	Mentor Graphics	Developer	
Mika Nuotio	Ericsson	Developer	
Patti Rusher	EIA/EDIF	Standards	
Lutz Treutler	FED		Review Only
Tom Vandenberge	Texas Instruments	User	Review
Nikolay Vitsyn	Independent	Developer	Review
Khan Vu	Sun	CAD Integrator	
John Welsh	Lockheed Martin ATL	CAD Integrator	
Jim Wilmore	HP	CAD Integrator	Review

Table A.3: Technology Libraries and Models Working Group (TLM)

Name	Representing	Background	Participation
Ron Waxman	Independent Consultant	Standards	Co-Chair
Steve Schulz	Texas Instruments	User	Vice-Chair
Dieter Bergman	IPC	Standards	Review
Dennis Brophy	Mentor Graphics	Developer	Review
Ranjit Chandra	Cadence	Developer	Review
Shirshen Chang	Synopsys	Developer	Review
Arindam Chatterjee	Texas Instruments	User	Review
Ron Christopher	Independent	CAD Integrator	Review
Don Cottrell	CFI	Standards	Review
Mike Emley	ViewLogic	Developer	Review
John Evans	Lockheed Martin	CAD Integrator	Review
Shahram Farnazadeh	Georgia Tech		
Elisa Finnie	Independent	Developer	Review
Rob Fletcher	IEEE	Standards	Review
Donna Fritz	EDAC		
Ian Getreu	Analogy	Developer	Review
Rich Goldman	EDAC, Synopsys	Developer	Review
Steve Grout	SEMATECH	User	Review
Carl Hein	Lockheed Martin	CAD Integrator	Review
Mitch Heins	CFI	CAD Integrator	Review
Shanker Hemmady			
Jan Johansson	Ericsson	User	Review Only
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	Review Only
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
Mike Krause	Nortel	User	Review

Table A.3: Technology Libraries and Models Working Group (TLM)

Name	Representing	Background	Participation
Pat McHug	Army Research Labs	Government	Review
Mika Nuotio	Ericsson	Developer	Review
Hank Oredson	Mentor Graphics	Developer	Review
Harry Parkinson	DEC	User	Review
Susan Runowicz-Smith	LSI Logic	User	Review
Patti Rusher	EIA/EDIF	Standards	Review
William R. Simpson	Institute for Defense Analysis	User	Review
John Teets	CFI	Standards	Review
Yatin Trevedi			Review
Tom Vandenberg	Texas Instruments	User	Review
Steve Waterbury	NASA	User	Review
John Welsh	Lockheed Martin ATL	CAD Integrator	Review
Hitoshi Yoshizawa	NEC	User	Review